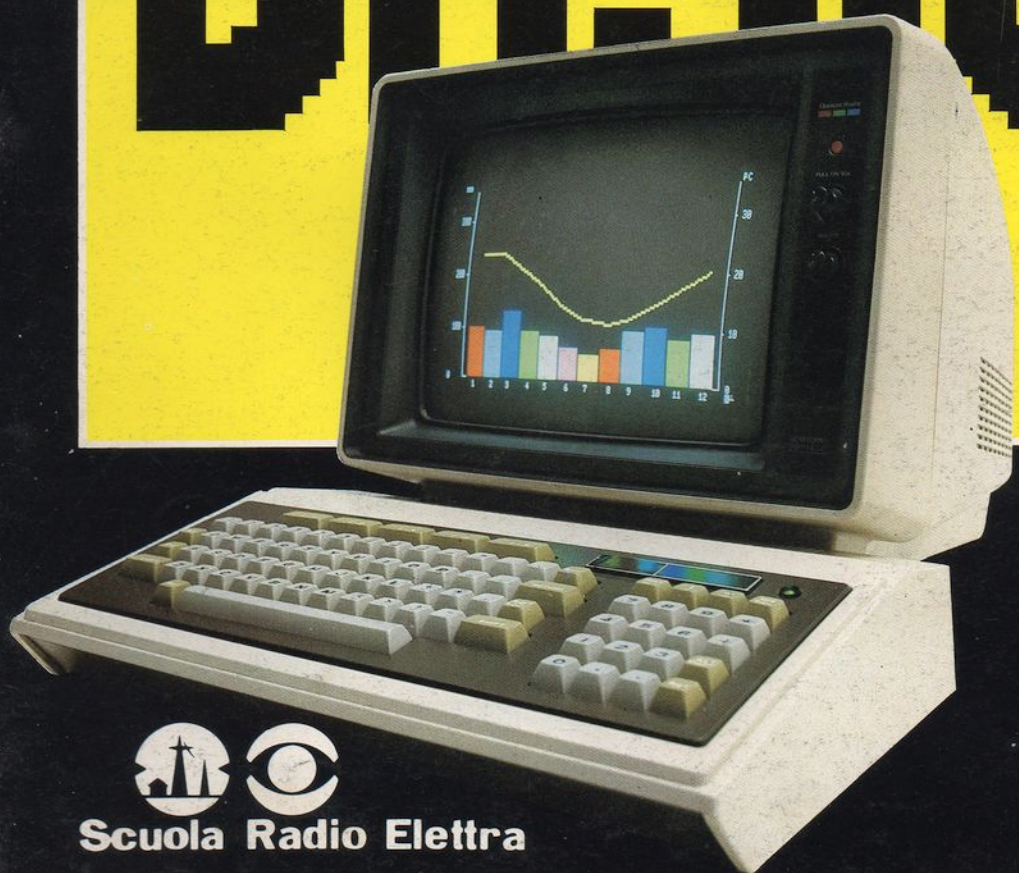
PARLA BASIC



Scuola Radio Elettra

Scuola Radio Elettra
Torino

Analisi e programmazione

Lezione 12

1 — Memoria principale e memorie ausiliarie
2 — L'archivio o file d'informazioni
3 — Organizzazione delle informazioni sull'unità a disco
4 — Organizzazione delle informazioni sull'unità a nastro
5 — Altri criteri di gestione delle informazioni nella memoria ausiliaria
6 — Le transazioni tra memoria principale e memoria secondaria

Che cosa apprenderà

Nella presente lezione verranno trattati i concetti inerenti la gestione delle informazioni, descrivendo le procedure adottate per trasferire file di dati dalla memoria principale a quella ausiliaria e viceversa.

Si completerà, inoltre, l'esame delle caratteristiche presentate dalla memoria principale del computer e dalla sua memoria ausiliaria (disco o nastro magnetico).

La memoria ausiliaria assume un'importanza fondamentale per l'archiviazione dei dati; senza di essa, infatti, il computer non potrebbe svolgere tutte quelle svariate applicazioni che hanno reso possibile la sua diffusione.

1. - Memoria principale e memorie ausiliarie

Come già precisato nelle precedenti lezioni, la memoria principale o interna del calcolatore è costituita da un insieme di celle elementari o locazioni, ognuna delle quali può contenere un byte (ossia 8 bit).

Il numero totale dei byte memorizzabili dipende dal tipo di calcolatore e varia in ragione di multipli di 1024.

Infatti, 1024 byte corrispondono a 1 Kbyte (kilobait), il quale viene anche indicato con la sola lettera K, che è l'unità di misura della capacità della memoria.

Un computer avente quindi una capacità di 48 K di memoria principale comprende 48 × 1024 celle o byte elementari, cioè 49152 byte.

Per potere individuare le varie locazioni di memoria, allo scopo di immettere o prelevare da esse i dati, tali locazioni sono numerate da Ø a 65535 in decimale, oppure da ØØØØ a FFFF in esadecimale. Gli indirizzi espressi in quest'ultima notazione sono normalmente preceduti dal carattere $.

In questo modo il numero che contraddistingue ciascuna locazione costituisce l'**indirizzo** mediante il quale si può facilmente accedere ad essa.

La memoria del computer **è organizzata in pagine**, ognuna delle quali può contenere 256 locazioni indirizzate da Ø a 255.

Facendo il prodotto 256*256 si ottiene 65536, che è appunto il numero di tutte le possibili locazioni di memoria, come già precisato.

I programmi ed i dati oggetti dell'elaborazione, rappresentati internamente al computer in codifica binaria ed esternamente in caratteri simbo-

lici, vengono memorizzati nell'area specifica di memoria ad accesso casuale (RAM). Questa può essere paragonata ad una lavagna sulla quale si può scrivere, leggere e cancellare quante volte si desidera.

La memorizzazione della memoria principale (RAM) non è permanente bensí la sua durata è limitata al solo tempo di elaborazione, per cui in questo caso la memoria viene anche definita **memoria volatile**.

Come la durata, pure la capacità di memorizzazione è limitata, mentre l'accessibilità alla singola cella è notevole, tant'è che la memoria principale si definisce anche **memoria ad accesso veloce**.

Alle caratteristiche ora descritte, relative alla memoria interna principale del computer, si contrappongono quelle delle memorie ausiliarie o di massa (memoria a disco e memoria a nastro). Queste memorie sono dotate di grande capacità di memorizzazione, ma hanno una lenta accessibilità alla singola informazione.

Notevoli progressi sono stati compiuti negli ultimi anni per migliorare da una parte la capacità delle memorie interne e dall'altra la velocità di accesso delle memorie ausiliarie.

Da un punto di vista costruttivo, la memoria principale è molto piú costosa e complessa di una memoria ausiliaria, perché tutte le celle di memoria sono da interconnettere tra loro sia fisicamente sia logicamente, il che richiede collegamenti, oltre che tra cella e cella, anche tra queste ultime e l'unità logico-aritmetica (o il microprocessore, nel caso dei personal computer).

Al contrario, le memorie ausiliarie sono costituite essenzialmente da semplici supporti magnetici, su cui sono scritte, in modo piú o meno organizzato, le informazioni che poi dovranno essere reperite con l'ausilio di programmi di ricerca (come i DATA BASES MANAGEMENT SYSTEMS, sistemi informativi di gestione dei dati).

I programmi di ricerca (anche se potenti e sofisticati) per il reperimento di una singola informazione possono operare "solo logicamente" sui flussi di dati (i file), eseguendo confronti ripetuti tra la massa dei dati ed il dato singolo oggetto di ricerca.

Al fine di chiarire meglio la differenza tra i sistemi di gestione delle informazioni della memoria principale e i sistemi della memoria ausiliaria, facciamo un semplice esempio.

L'individuazione di un elemento contenuto nella memoria principale si ottiene prontamente citando nel programma il nome di quell'elemento (seguito eventualmente da parametri o indici, nel caso si tratti di elemento di matrice). Il sistema di controllo dell'unità centrale (CPU), durante l'esecuzione del programma, attiva il circuito elettronico di confronto, il quale, per raffrontare il nome dell'elemento da ricercare con tutti quelli presenti nella memoria principale, impiega un tempo brevissimo, dell'ordine dei microsecondi (milionesimi di secondo).

L'individuazione di un elemento d'informazione in un file registrato su nastro o disco è ottenuta invece attraverso una ricerca piú lenta effettuata da appositi programmi che sono costretti ad operare **raffronti indiretti** tra l'elemento da ricercare e i vari elementi del file stesso.

Si definisce raffronto indiretto tra due elementi un confronto bit per bit eseguito su un supporto diverso da quello in cui essi sono registrati. Nel caso infatti di elementi registrati su memorie ausiliarie, prima del confronto questi elementi vengono trasferiti nella memoria principale e poi comparati bit per bit tra loro (*fig. 1*).

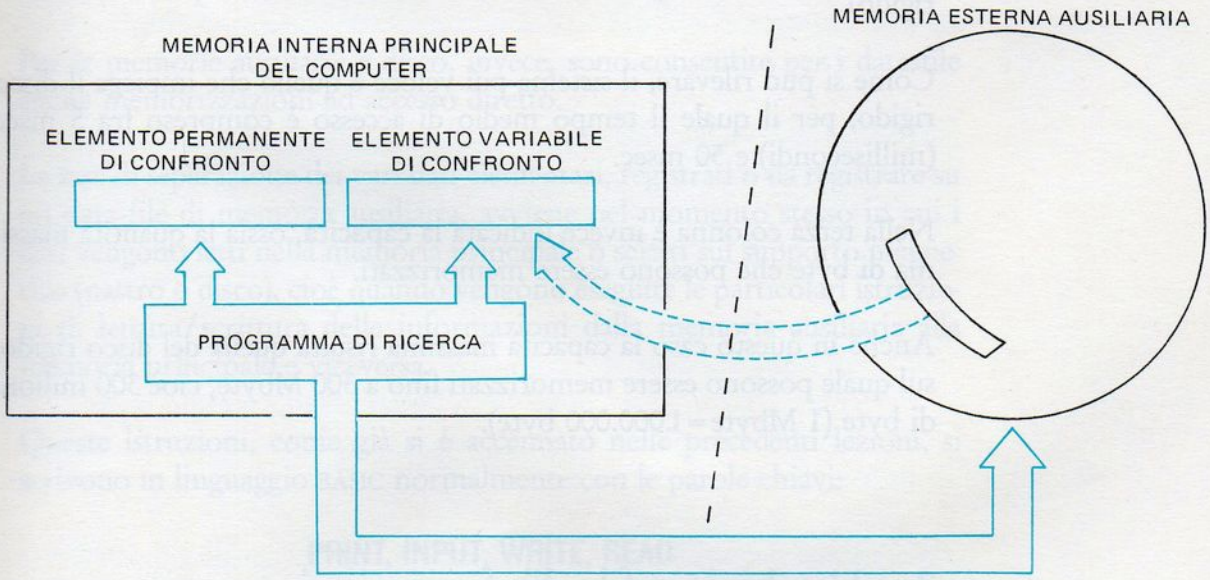


Fig. 1 – Ricerca di un elemento d'informazione memorizzato nella memoria esterna del computer.

In altre parole, la ricerca avviene mediante continui raffronti tra un primo elemento permanente memorizzato nella memoria principale ed un secondo elemento (variabile), inviato continuamente dalla memoria ausiliaria alla stessa memoria principale.

Nella tabella della *fig. 2* sono riportate le caratteristiche piú significative dei principali sistemi di registrazione magnetica dei dati impiegati nelle memorie ausiliarie dei personal computer.

SISTEMI DI REGISTRAZIONE	TEMPO MEDIO DI ACCESSO	CAPACITA'
CASSETTA MAGNETICA	Da alcune decine di secondi a 30 minuti	Da 10 a 100 Kbyte
FLOPPY-DISK	Da 50 a 200 msec	Da 100 Kbyte a 5 Mbyte
HARD-DISK	Da 5 a 50 msec	Da 5 a 300 Mbyte

Fig. 2 – Caratteristiche delle tecniche di registrazione magnetica utilizzate nelle memorie ausiliarie dei personal computer.

Nella seconda colonna di tale tabella è riportato **il tempo medio di accesso**, ossia il tempo occorrente per individuare un dato sul nastro magnetico racchiuso in cassetta o su floppy-disk o su hard-disk (disco rigido).

Come si può rilevare, il sistema piú veloce è quello che impiega il disco rigido, per il quale il tempo medio di accesso è compreso fra 5 msec (millisecondi) e 50 msec.

Nella terza colonna è invece indicata la **capacità**, ossia la quantità massima di byte che possono essere memorizzati.

Anche in questo caso la capacità massima risulta quella del disco rigido, sul quale possono essere memorizzati fino a 300 Mbyte, cioè 300 milioni di byte (1 Mbyte = 1.000.000 byte).

2. – L'archivio o file d'informazioni

Come già precisato nella lezione *Analisi e programmazione 8*, il file o archivio d'informazioni è un raggruppamento organizzato e strutturato di informazioni (numeriche ed alfanumeriche) e costituisce la forma logica principale di memorizzazione dei supporti fisici di memoria ausiliaria (dischi e nastri magnetici).

File programma e file-dati

Un file di dati può essere costituito da istruzioni di programma, nel qual caso si chiama **file-programma**, oppure da dati propriamente numerici e alfanumerici, nel qual caso si chiama **file-dati** (program-files e data-files, in inglese).

Un data-file può essere, al suo interno, strutturato in modo da presentare suddivisioni ulteriori (campi, sottocampi, record, ecc.) e "accessibile" in due modi diversi: **in modo sequenziale** e **in modo diretto** o **random.**

Per il primo modo è sufficiente che il tipo di memorizzazione sul supporto di massa sia sequenziale, cioè che le informazioni siano "registrate" una di seguito all'altra, eventualmente separate da speciali caratteri separatori; per il secondo modo di accesso (diretto o random), è necessario che le singole informazioni siano raggruppate in record, essendo ogni record costituito da un numero uguale e fisso di byte ed eventualmente suddiviso, a sua volta, in campi, ognuno dei quali è separato dall'altro da speciali caratteri separatori.

Nel caso di memoria ausiliaria a nastro magnetico, esiste per i data-file soltanto un tipo di memorizzazione: il modo sequenziale.

Per le memorie ausiliarie a disco, invece, sono consentite per i data-file anche memorizzazioni ad accesso diretto.

La fase di separazione dei vari dati elementari, registrati o da registrare su un data-file di memoria ausiliaria, avviene nel momento stesso in cui i dati vengono letti nella memoria principale o scritti sul supporto magnetico (nastro o disco), cioè quando vengono eseguite le particolari istruzioni di lettura/scrittura delle informazioni dalla memoria ausiliaria alla memoria principale e viceversa.

Queste istruzioni, come già si è accennato nelle precedenti lezioni, si scrivono in linguaggio BASIC normalmente con le parole chiavi:

PRINT, INPUT, WRITE, READ

seguite da speciali caratteri di controllo o speciali caratteri dipendenti dal particolare sistema personal computer che si ha a disposizione.

Sul personal APPLE II, ad esempio, per registrare su un data-file sequenziale di floppy-disk le seguenti tre parole (ognuna su tre campi diversi): "SCUOLA", "RADIO ELETTRA", "TORINO" si utilizzano le istruzioni BASIC riportate di seguito.

```
10 D$=CHR$(4):REM*D$ CONTIENE CARATTERE DI CONTROLLO*
20 PRINT D$;"OPEN SCUOLA"
30 PRINT D$;"WRITE SCUOLA"
40 PRINT "SCUOLA"
50 PRINT "RADIO ELETTRA"
60 PRINT "TORINO"
70 PRINT D$;"CLOSE SCUOLA"
80 END
```

Apertura del file

Come si vede, la sintassi BASIC prevede che ogni file, prima di essere registrato o letto, sia "aperto" con un'istruzione apposita chiamata **OPEN**, come se si trattasse di un libro preso da una libreria per essere consultato.

Chiusura del file

Analogamente, dopo che si è effettuata la registrazione o lettura, il file di dati deve essere "chiuso" con l'istruzione **CLOSE**.

L'istruzione 3Ø viene scritta dopo l'apertura del file e prima di ogni operazione di scrittura/lettura. La parola chiave **WRITE** serve a predisporre la testina di lettura/scrittura dell'unità magnetica di memoria ausiliaria nella posizione iniziale del file, in modo che proprio da questo punto inizino le registrazioni delle informazioni. Queste ultime sono inviate su disco attraverso le successive istruzioni 4Ø÷6Ø.

In questo caso la parola chiave **PRINT** ha un significato tutto particolare; se è scritta dopo la **WRITE**, serve non già a visualizzare su schermo monitor o TV oppure su stampante, bensí ad effettuare un'operazione di registrazione su un supporto di memoria ausiliaria.

Analogamente, per eseguire operazioni di lettura dalla memoria ausiliaria alla memoria principale del computer viene utilizzata, dopo le parole chiavi **OPEN** e **READ**, l'istruzione BASIC **INPUT**, seguita da uno o piú nomi di locazioni di memoria dove andranno a finire le informazioni registrate sul supporto di memoria ausiliaria.

Altri sistemi personal computer utilizzano, dopo le parole chiavi **PRINT** e **INPUT**, il carattere speciale #, seguito da un numero intero (detto anche numero di file o "number file") associato al file da registrare o leggere.

Ad esempio, se al momento dell'apertura del file-dati chiamato "CLIENTI" si vuole associare a questo file il number-file 1, si scriverà l'istruzione che segue:

OPEN "I", #1,"CLIENTI"

Successivamente, per leggere un campo da questo file-dati si scriverà:

INPUT # 1, A$

dove A$ è il nome della locazione di memoria principale in cui sarà immessa l'informazione (contenuta in un campo) registrata sul supporto di memoria ausiliaria (disco o nastro).

Se si vogliono immettere in memoria piú campi di informazioni, occorre scrivere l'istruzione **INPUT** seguita da un elenco appropriato di nomi di locazioni di memoria, nomi che devono rispettare le regole del linguaggio di programmazione BASIC.

La separazione dei valori d'informazione avviene proprio nel momento in cui vengono eseguite le istruzioni **PRINT** (per la registrazione) e **INPUT** (per l'immissione o lettura in memoria principale).

Altri sistemi personal computer, come l'M20 Olivetti e IBM PC utilizzano le speciali istruzioni **GET** e **PUT** per effettuare operazioni di lettura e scrittura di informazioni tra la memoria ausialiaria e la memoria principale.

Inoltre, altre speciali istruzioni, tipo l'istruzione **FIELD**, consentono di "strutturare" i dati (secondo particolari organizzazioni) prima che questi possano essere memorizzati in uno dei due supporti di memoria (principale e secondaria).

Riassumendo quanto precedentemente asserito, le informazioni che possono essere memorizzate nella memoria ausiliaria sono di due tipi:

1) **programmi**
2) **dati**

Entrambi sono organizzati in file (flussi d'informazione) e sono differentemente gestiti dal computer.

I programmi (detti anche program-file) sono gestiti con speciali comandi BASIC di lettura/scrittura, quali ad esempio i comandi **LOAD** e **SAVE** che, come è noto, servono rispettivamente ad immettere programmi dalla memoria ausiliaria nella memoria principale ed a registrare programmi dalla memoria principale in quella ausiliaria.

I dati-file sono gestiti esclusivamente dall'interno di un programma con speciali istruzioni, tipo **INPUT #1** e **PRINT #1**, che come nel caso dei programmi servono ad effettuare la lettura e la scrittura di valori numerici e stringa. Nel caso specifico dei data-file, però, le istruzioni di lettura/scrittura sono in un certo senso contornate da altre istruzioni, che servono a migliorare l'efficienza operativa del sistema personal computer.

In particolare, prima di effettuare la lettura di un data-file, occorre che il sistema predisponga l'organo di lettura/scrittura dell'unità ausiliaria all'inizio del file stesso, utilizzando l'istruzione di apertura **OPEN**, seguita dal nome del file di dati su cui si vuole operare e da altri parametri.

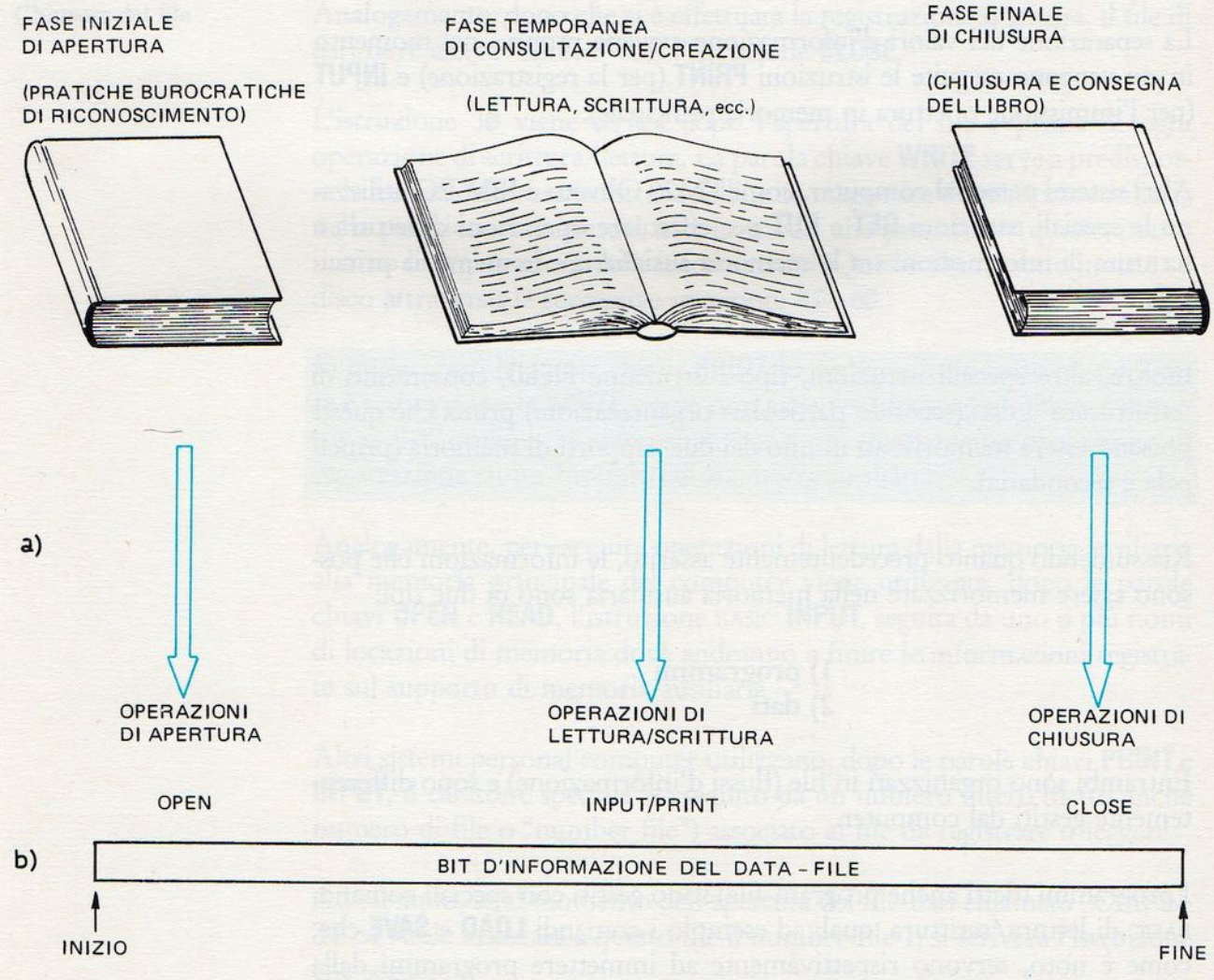


Fig. 3 - Analogie tra l'apertura, la lettura e la chiusura di un libro (a), e le corrispondenti fasi operative eseguite su un file di dati (b).

Parimenti, dopo avere eseguito le operazioni di lettura e scrittura su un certo file, occorre che i dati ivi memorizzati siano e rimangano protetti da eventuali errori accidentali, che potrebbero essere provocati da qualche elaborazione.

La protezione del file appena elaborato è ottenuta tramite l'istruzione **CLOSE**.

In pratica, tutto si svolge come quando si va in una biblioteca per consultare un libro (*fig. 3-a*); prima di leggere l'informazione cercata su quest'ultimo, è necessario effettuare a parte le procedure preliminari, consistenti in una vera e propria operazione di apertura: si apre cioè il libro e se ne legge il titolo in prima pagina per accertarsi che sia quello desiderato. Dopo avere letto l'informazione occorre poi "chiuderlo" prima di depositarlo nello scaffale.

Nella *fig. 3-b* sono schematizzate le corrispondenti fasi operative di apertura, lettura/scrittura e chiusura del file.

3. – Organizzazione delle informazioni sull'unità a disco

Come si è visto nel precedente capitolo ed in base alle osservazioni di carattere generale riportate nella lezione *Analisi e programmazione 8*, la memoria ausiliaria a disco costituisce il migliore supporto per l'archiviazione delle informazioni, soprattutto nel caso in cui queste debbano essere rese "disponibili" prontamente; per l'archiviazione di dati storici è sufficiente invece servirsi di memorie a nastro che, pur essendo lente, sono particolarmente "capaci" di contenere quantità elevate di dati.

Nel campo dei sistemi personal computer, la memoria a disco è generalmente costituita da uno o due piccoli contenitori (drive), nei quali sono inseriti i supporti fisici della memoria ausiliaria: i cosiddetti floppy-disk, consistenti in dischetti flessibili di plastica di dimensioni contenute (da 3 a 12 pollici di diametro) e rivestiti di materiale ferro-magnetico.

Le informazioni (dati numerici e stringa, programmi codificati in linguaggi simbolici, linguaggi macchina) sono registrate su una o su entrambe le superfici superiore ed inferiore del floppy-disk, lungo minutissimi solchi detti **piste** o **tracce**.

I floppy-disk da 5 pollici di diametro sono generalmente costituiti da 35

piste per facciata, numerate da Ø a 34 e suddivise in **settori**.

Ogni disco, prima di essere utilizzato per la registrazione delle informazioni, va inizializzato dal sistema operativo per dischi (DOS), contenuto nel computer oppure nel drive (come nei modelli della Commodore), mediante appositi comandi BASIC. Per esempio, per inizializzare un disco il computer APPLE II usa il comando **INIT**; altri sistemi utilizzano il comando **FORMAT**.

Comandi **INIT** e **FORMAT**

L'inizializzazione o formattazione consiste nello strutturare le piste del disco in un numero fisso di **settori** e nel separare questi ultimi con zone magnetiche "vuote", dette GAP.

Ogni settore è suddiviso a sua volta in **blocchi**, ciascuno dei quali può immagazzinare sino a 256 byte di dati (come nella memoria principale del computer il byte del disco è costituito da 8 bit).

Occorre tenere presente che la formattazione cancella qualsiasi precedente informazione presente sul disco. Pertanto, prima di procedere alla formattazione di un disco già registrato, occorre **accertarsi che il suo contenuto possa essere effettivamente cancellato.**

Alcuni sistemi personal computer definiscono la lunghezza massima di un record, in un file ad accesso diretto, pari a 256 byte; altri sistemi invece consentono record di lunghezza compresa fra 1 byte e 4096 byte.

Nella *fig. 4* è rappresentata in forma schematica l'organizzazzione dei data-file sulla traccia di un floppy-disk.

Come si può notare, i vari settori sono tra loro separati da spazi non registrati (GAP) e preceduti da un'intestazione (header, in inglese), registrata durante la formattazione, nella quale sono contenuti il numero della traccia, il numero del settore ed eventuali altre informazioni.

Alcune delle 35 piste di ogni facciata del disco sono riservate al sistema operativo ed utilizzate da quest'ultimo per la gestione dei file registrati sullo stesso disco (i file possono essere sia di programmi sia di dati).

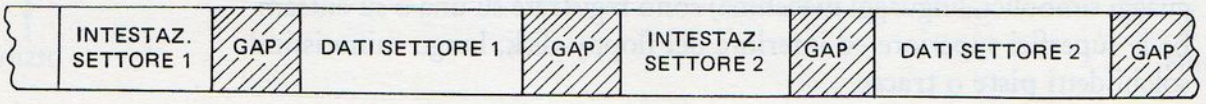


Fig. 4 - Schema dell'organizzazione dei data-file sulla traccia di un floppy-disk.

Una di queste piste contiene l'elenco PISTA-SETTORE, consultando il quale il sistema operativo è in grado di rintracciare rapidamente tutti i record o campi di informazione che fanno parte di un certo file.

Un'altra pista contiene l'elenco o il catalogo (directory) di tutti i file registrati sul disco. Tale elenco, per la maggior parte dei personal computer, si può configurare come nello schema della *fig. 5*.

TIPO	SETTORI OCCUPATI	DESCRIZIONE
A	ØØ2	PRODOTTO MATRICI
*B	ØØ3	CALCOLO RADICI EQUAZ.
A	ØØ4	DISEGNO 1
A	ØØ2	PROGR. UTILITA'
A	ØØ8	FATTURAZIONE
*B	Ø14	ROUTINE DI CALCOLO
I	Ø1Ø	COMBINAZIONI
T	ØØ2	DATI 1
*T	ØØ4	DATI 2
T	ØØ4	DATI 1984
T	ØØ8	DATI 3
A	Ø2Ø	PROGR. COSTI
A	ØØ2	PROGR. RICAVI

Fig. 5 – Tipico prospetto contenente l'elenco e la descrizione dei file registrati su un disco.

La lettera T riportata in tale figura sta ad indicare che il file relativo è del tipo "Testo", cioè contiene dati (numerici o stringa).

La lettera A sta a significare che il file è un programma scritto nel linguaggio BASIC speciale **Applesoft** (nome che si dà al linguaggio interprete BASIC del Personal APPLE II).

La lettera I indica invece un programma scritto nel linguaggio **Integer** BASIC (una specie di BASIC simile all'Applesoft, ma che consente di operare su numeri interi).

La lettera B, infine, sta ad indicare che il programma è scritto in linguaggio macchina.

Nella seconda colonna della *fig. 5* sono elencati dei numeri interi che rappresentano l'occupazione, espressa in settori, di ciascun file sul disco. Nell'ultima colonna sono poi riportati i nomi dei file, cosí come sono stati registrati sul disco.

Il carattere *, disposto alla sinistra di alcune lettere della prima colonna, indica che il file relativo è "bloccato", cioè è protetto da eventuali operazioni di scrittura.

Comando LOCK

Normalmente si blocca un file con appositi comandi BASIC, come ad esempio il comando **LOCK**, perché si desidera preservarlo da eventuali manomissioni accidentali, data la sua importanza.

Comandi DIR, CAT, CATALOG

Per la lettura dei nomi dei file presenti nell'indice del disco vengono utilizzati i seguenti comandi BASIC: **DIR**, **CAT**, **CATALOG**, a seconda del tipo di personal computer.

Comandi KILL e DELETE

Infine, per cancellare il nome di un file viene usato, per esempio, il comando **KILL** oppure il comando **DELETE**.

Come già precisato, oltre ai floppy-disk vengono pure impiegati i dischi rigidi, denominati anche **Winchester**, dato il tipo di tecnologia adottata per la loro costruzione.

Questi tipi di dischi hanno un grandissimo potenziale di memoria, ma, essendo molto costosi, vengono utilizzati esclusivamente per applicazioni professionali.

4. – Organizzazione delle informazioni sull'unità a nastro

L'organizzazione dei file sull'unità a nastro, sia che si tratti di programmi sia che si tratti di dati o piú propriamente di informazioni numeriche e stringa, è soltanto di tipo **sequenziale** o **seriale**.

Il file registrato su nastro può essere strutturato anche in record, ognuno dei quali individuato da campi informativi denominati "CHIAVI", ed ogni record può essere strutturato in ulteriori sottocampi.

L'accessibilità e la ricerca delle singole informazioni sono comunque sempre di tipo sequenziale. Per questo motivo l'unità a nastro si può considerare un supporto di memoria ausiliaria meno veloce dell'unità a disco.

Come già precedentemente accennato, l'unità a nastro viene molto utilizzata per l'archiviazione di notevoli quantità di dati, che però debbono essere riutilizzati solo saltuariamente o periodicamente; per esempio, dati di vendite mensili di un'azienda, dati di contabilità per la fatturazione mensile, per la retribuzione del personale, ecc., tutti dati che comportano elaborazioni a scadenze fisse: fine settimana, fine mese, fine anno.

Il calcolo delle retribuzioni mensili dei dipendenti di un'azienda, per esempio, è una tipica applicazione adatta ad essere elaborata con memorie sequenziali sull'unità a nastro.

Le informazioni relative ad ogni dipendente, infatti, possono essere organizzate e memorizzate nel seguente ordine sequenziale:

a) - codice dipendente
b) - nominativo
c) - indirizzo
d) - categoria sindacale
e) - paga base
f) - ore lavorate
g) - ore straordinarie
h) - altre informazioni

e quindi possono essere registrate su un nastro, una di seguito all'altra (eventualmente seguendo valori di codici crescenti).

Il programma ora considerato, che elabora le paghe, ricercherà sul nastro il "primo" CODICE DIPENDENTE ed invierà nella memoria principale dell'elaboratore le informazioni registrate successivamente (NOMINATIVO, INDIRIZZO, CATEGORIA SINDACALE, ecc.). Quindi, eseguirà una semplice contabilizzazione delle ore lavorate (moltiplicando il monte ore per la PAGA ORARIA, ed eventualmente aggiungendo le ore straordinarie). Infine, provvederà ad effettuare una stampa, o visualizzazione, del prospetto paga su appositi moduli già predisposti sulle unità periferiche di stampa.

L'informazione minima che può essere letta o scritta su un supporto di memoria ausiliaria (nastro o disco) è il carattere (che, come sappiamo, occupa un byte).

Piú caratteri possono essere raggruppati per formare un campo e piú campi possono formare un record.

Nella maggior parte dei personal computer, che utilizzano memorie ausiliarie a nastro, non è previsto l'utilizzo dei file organizzati in record. In questi casi l'informazione minima è quindi il carattere o i gruppi di caratteri (campi), ognuno separato dal successivo da uno speciale carattere separatore (generalmente viene utilizzato il carattere ↵ o RETURN o CR, che viene registrato con il codice ASCII decimale 13).

Il sistema per l'archiviazione dei programmi e dei dati sui personal computer e sugli home computer che utilizzano le memorie o nastri impiega i comuni registratori a cassetta.

A questo proposito è buona norma usare sempre cassette di buona qualità e di breve durata, ad esempio i tipi C30, che consentono 15 minuti di registrazione per lato; ciò perché il nastro magnetico delle cassette a lunga durata, come le C90 e le C120, essendo piú sottile di quello adottato dalle C30, è maggiormente soggetto ad alterazioni (stiramenti), che possono causare serie difficoltà nelle operazioni di caricamento e salvataggio dei programmi.

5. – Altri criteri di gestione delle informazioni nelle memorie ausiliarie

Come si è visto precedentemente, la gestione delle informazioni numeriche o alfanumeriche (file-dati), memorizzate in un supporto di memoria ausiliaria, viene normalmente effettuata attraverso le seguenti tre fasi:

1ª - apertura del file-dati;
2ª - operazioni di lettura/scrittura;
3ª - chiusura del file-dati.

Questi tre tipi di operazioni vengono "programmati" attraverso speciali istruzioni BASIC, che si differenziano da sistema a sistema.

Le informazioni, invece, costituite da programmi (file-programmi), sono gestite attraverso l'esecuzione immediata di speciali comandi BASIC, quali i comandi:

SAVE
LOAD
VERIFY
LIST

Queste regole sono valide per la maggior parte dei personal computer.

Per i sistemi piú potenti, dotati di linguaggi BASIC piú sofisticati, è possibile leggere o registrare i file-dati con gli stessi comandi con cui si leggono o si registrano i file-programmi.

I dati numerici o alfanumerici sono cioè organizzati mediante istruzioni BASIC con propri numeri di riferimento.

Di seguito è riportato un gruppo di dati, organizzato a guisa di istruzioni:

```
10 "NUMERO PI-GRECO" 3.14159
20 "NUMERO NEPERIANO" 2.78
30 "COSTANTE DIELETTRICA"
40 . . . . . .
50 . . . . . .
60 . . . . . .
```

In questo elenco sono indicati, in corrispondenza di ogni numero di riferimento, due dati:

1) - una stringa (tra virgolette) che serve a commentare il dato riportato successivamente;

2) - un numero decimale.

Tra il dato stringa 1) ed il valore numerico 2) c'è uno spazio, che serve a separare un dato dall'altro.

Un file di dati, come quello ora considerato, si gestisce allo stesso modo dei file-programmi, utilizzando ad esempio il comando BASIC:

SAVE

per salvarlo (cioè registrarlo) su un supporto di memoria ausiliaria; il comando:

LOAD

per caricarlo da un supporto di memoria ausiliaria alla memoria principale; il comando:

LIST

per listarlo sullo schermo, ecc.

6. – Le transazioni tra memoria principale e memoria secondaria

Il buffer

Quando da programma viene eseguita un'istruzione di scrittura/lettura, il valore numerico o stringa staziona per un certo tempo in un'area speciale di memoria, detta BUFFER.

Quest'area, costituita da un certo numero di celle elementari, serve appunto ad immagazzinare **temporaneamente** tutte le informazioni che vengono scambiate tra la memoria principale e quella secondaria.

L'effettiva operazione di lettura/scrittura avviene nel momento stesso in cui l'area di buffer risulta "riempita" di dati.

Lo schema delle transazioni è rappresentato nella *fig. 6*.

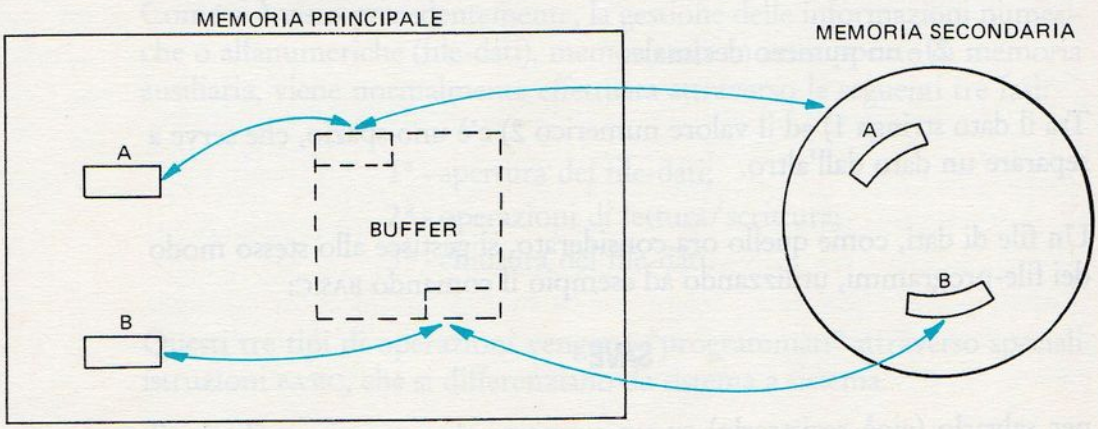


Fig. 6 – Area di buffer per le transazioni delle informazioni provenienti dalla o dirette alla memoria principale.

Normalmente, il buffer di un personal computer è costituito da 512 byte di memoria, dei quali 256 sono riservati per immagazzinare informazioni che transitano dalla memoria principale alla memoria secondaria e gli altri 256 per immagazzinare informazioni che transitano in senso opposto.

L'area di memoria riservata al buffer viene definita nel momento stesso in cui si effettua l'operazione di apertura di un file-dati, come di seguito riportato:

```
OPEN # n, "file-nome".
```

Nell'istruzione in questione, **n** rappresenta un numero intero (compreso generalmente tra 1 e 16), associato ad un'area di buffer attraverso cui transiteranno le informazioni provenienti o dirette al file-dati avente il nome "file-nome".

Nelle successive operazioni di lettura/scrittura del file-dati comparirà il riferimento **n** al posto del nome del file-dati.

Per esempio, scrivendo successivamente:

```
10 OPEN # 1, "DATI"
15 INPUT # 1, A
```

si predisporrà il sistema in modo da far leggere dal file-dati di nome "DATI" un dato numerico e da inviare questo dato nella memoria principale nella locazione A, dopo averlo fatto transitare nell'area di buffer # 1.

Analogamente, per le operazioni di scrittura impartendo le istruzioni:

```
OPEN # 2, "ELENCO"
PRINT # 2, A$, B$, C$
```

si predisporrà il sistema ad aprire un file-dati di nome "ELENCO" con associato il buffer # 2 ed a scrivere sul file stesso i valori stringa contenuti nelle locazioni di memoria A$, B$, C$, dopo averle fatte transitare nel buffer # 2.

* * * * * * *

Da memorizzare

- Il numero che contraddistingue ciascuna locazione ne costituisce l'**indirizzo** mediante il quale si può facilmente accedere ad essa.
- La memoria del computer è organizzata in **pagine**, ognuna delle quali può contenere 256 locazioni indirizzate da **0** a 255.
- La memoria principale viene definita **ad accesso veloce**, ma la sua capacità di memorizzazione è limitata.
- Le memorie ausiliarie o di massa hanno **grande capacità di memorizzazione**, ma presentano una lenta accessibilità alla singola informazione.
- Nella memoria ausiliaria la ricerca dei dati memorizzati avviene per continui raffronti tra un primo elemento permanente, memorizzato nella memoria principale, ed un secondo elemento (variabile), inviato continuamente dalla memoria ausiliaria alla stessa memoria principale.
- Un file di dati può essere costituito da istruzioni di programma (**file-programma**), oppure da dati propriamente numerici e alfanumerici (**file-dati**).
- Per effettuare operazioni di lettura/scrittura di un file, occorre prima impartire il comando di apertura (**OPEN**), seguito dal nome del file di dati su cui si vuole operare.
- Dopo avere eseguito le operazioni di lettura e scrittura di un file, questo deve essere chiuso (**CLOSE**), per fare in modo che i dati in esso memorizzati risultino protetti da eventuali errori accidentali.
- Ogni disco, prima di essere utilizzato, deve essere inizializzato dal sistema DOS mediante appositi comandi BASIC, quali i comandi **INIT** e **FORMAT**.
- L'inizializzazione o formattazione del disco ha lo scopo di suddividere le tracce di registrazione in **settori**, a loro volta ripartiti in **blocchi**, ognuno dei quali può contenere sino a 256 byte di dati.
- Per preservare un file su disco da eventuali manomissioni accidentali si usa il comando BASIC **LOCK**.
- Per la lettura dei nomi dei file contenuti nell'indice del disco vengono utilizzati i seguenti comandi BASIC **DIR**, **CAT**, **CATALOG**, a seconda del tipo di personal computer.
- L'organizzazione dei file su disco può essere **sequenziale** e **ad accesso diretto**, mentre l'organizzazione dei file su nastro magnetico è **solo sequenziale**.
- I sistemi personal computer che utilizzano linguaggi BASIC piú sofisticati consentono di leggere o registrare i file-dati con gli stessi comandi con cui si leggono o si registrano i file-programmi.
- Il **buffer** ha il compito di immagazzinare temporaneamente le informazioni che vengono scambiate tra la memoria principale e quella secondaria. L'effettiva operazione di lettura/scrittura avviene nel momento stesso in cui l'area di buffer risulta riempita di dati.
- Il buffer di un personal computer è costituito da 512 byte di memoria.

Scuola Radio Elettra
Torino

Pratica del BASIC

Lezione 12

1 — La gestione dei dati sul registratore a cassette
2 — Esempio pratico di utilizzo di file-sequenziale
3 — Operatività dei tasti di funzione
4 — Mappe di memoria dei computer C64 e VIC-20

Che cosa farà

In questa lezione verificherà praticamente come si possono gestire i file-dati organizzati sequenzialmente su nastro magnetico.

Potrà così realizzare una procedura, costituita da due programmi, che consentirà di risolvere svariati problemi di tipo gestionale.

Inoltre, mediante appositi programmi esemplificativi, avrà modo di constatare l'operatività dei tasti di funzione.

Infine, a conclusione della lezione, verrà esaminata, in linea di massima, l'organizzazione delle diverse aree di memoria (mappe) dei computer C64 e VIC-20.

1. – La gestione dei dati sul registratore a cassette

Verificheremo ora, praticamente, come si può scrivere e leggere una serie di dati sull'unità a cassette, in modo da poterli utilizzare durante l'esecuzione di un programma o in qualsiasi altra circostanza in cui essi siano richiesti.

Per ottenere quanto precisato occorre creare un file, contenente i dati in questione, il quale, una volta registrato sul nastro, si possa leggere facilmente con un programma ed eventualmente modificare e riscrivere a seconda delle necessità.

I dati, come i programmi, vengono registrati sul nastro magnetico soltanto in modo sequenziale.

Per ogni operazione di scrittura o lettura delle informazioni occorre, in primo luogo (come già specificato nella lezione *Analisi e programmazione 12*), "aprire" un file-dati per comunicare al computer il proposito di utilizzare il registratore (oppure qualsiasi altra periferica).

Per fare ciò si ricorre all'istruzione BASIC:

OPEN.

La parola chiave **OPEN** è generalmente seguita da tre parametri numerici e dal nome del file-dati.

Il formato dell'istruzione **OPEN** è quindi il seguente:

OPEN A, B, C, "NOME"

La lettera A sta ad indicare **il numero del file scelto** (un numero intero positivo, che rappresenta il buffer di memoria attraverso il quale transitano le informazioni del file-dati). Questo numero, a cui faranno riferimento tutte le istruzioni successive, va da 1 a 255 (normalmente vengono però utilizzati soltanto i valori da 1 a 127).

Il secondo numero, B, sta ad indicare **il codice dell'unità ausiliaria** impiegata, in quanto il computer deve sapere a quale tipo di periferica deve rivolgersi, indipendentemente dal fatto che si tratti di una scrittura o di una lettura di dati.

Il numero B può assumere i valori di seguito elencati:

Ø per lo schermo;
1 per l'unità a cassette;
4 per la stampante;
8 per l'unità a disco.

Il terzo numero, C, sempre separato dal precedente da una virgola, costituisce **l'indirizzo secondario** che avverte il registratore di prepararsi a scrivere o a leggere i dati sul file.

Il numero C può assumere i tre seguenti valori:

C=Ø;
C=1;
C=2.

Se C=Ø, il file-dati è aperto per "leggere" delle informazioni. Quindi, la successiva istruzione BASIC sarà una **INPUT** # A (cioè un'immissione da nastro di un dato nella memoria principale, transitando per il buffer A), dove A è lo stesso numero del file utilizzato nell'istruzione **OPEN**.

Se C=1, il file-dati è aperto per "scrivere" delle informazioni su nastro. Quindi, la successiva istruzione BASIC sarà una **PRINT** # A (cioè un'emissione da memoria principale di un dato sul nastro, transitando per il buffer A).

Con C=1 il sistema provvede a registrare anche un carattere di fine-file (end-of-file) al termine dello stesso file, quando sia stata effettuata l'istruzione di chiusura **CLOSE**.

Occorre tenere presente che nelle istruzioni **INPUT** # e **PRINT** # il carattere # fa parte delle stesse parole chiavi e quindi non può essere scritto separato dai caratteri che lo precedono.

Se C=2, il file-dati viene sempre aperto in "scrittura", ma il sistema provvede, al momento della chiusura del file, a registrare un carattere di fine-nastro (end-of-tape) alla fine dei dati del file-dati.

Il carattere di fine-nastro impedisce alla testina magnetica dell'unità a nastro di proseguire oltre (esso costituisce quindi un limitatore fisico per le informazioni registrate su nastro).

Infine, con il termine "NOME" viene indicato il nome che si vuol dare al file; tale nome può anche essere omesso: in questo caso verrà letto il primo file trovato sul nastro.

Facciamo ora un esempio pratico di scrittura e lettura di un file-dati: supponiamo di volere registrare su nastro i primi 1ØØ numeri interi e successivamente di rileggere questi valori.

In un primo momento si tratta di creare un file-dati sequenziale su nastro, file costituito dai primi 1ØØ numeri interi, i quali possono essere creati dal programma stesso.

In un secondo momento, dopo avere avvolto manualmente la cassetta-nastro, si possono rileggere i dati numerici precedentemente creati per effettuare eventualmente sugli stessi ulteriori operazioni aritmetiche (ad esempio, l'elevazione al quadrato dei numeri letti e la stampa dei quadrati sullo schermo).

L'intera operazione può essere effettuata con un solo programma BASIC, (come riportato nella *fig. 1*), il quale in due fasi successive registra e legge da nastro una successione di numeri.

Come può rilevare, tra la prima e la seconda parte del programma è stata interposta un'istruzione **STOP**, in modo da consentire all'operatore di riavvolgere (sempre manualmente) il file-dati registrato su nastro.

Nell'istruzione 2Ø viene aperto (**OPEN**) il file-dati (a cui viene attribuito il numero di buffer 1) sul nastro magnetico (1) per la registrazione o la scrittura delle informazioni (1); il nome attribuito al file è "DATI 1-1ØØ".

L'istruzione 3Ø imposta un'iterazione che attribuisce alla variabile I i valori numerici compresi fra 1 e 1ØØ (corrispondenti ai dati del file), i quali vengono scritti, ossia registrati sul nastro, mediante l'istruzione **PRINT** # 1, I (linea di programma 4Ø). Il numero 1 presente in tale istruzione è lo stesso che è stato associato al nome del file nell'istruzione **OPEN**.

Al termine della scrittura dei dati, il file viene chiuso (**CLOSE** 1) mediante

```
10 REM*ESEMPIO DI REGISTRAZIONE*
15 REM*E DI LETTURA DA NASTRO*
20 OPEN 1,1,1,"DATI 1-100"
30 FOR I=1 TO 100
40 PRINT#1,I
50 NEXT I
55 CLOSE 1
60 STOP:PRINT"■"
70 OPEN 2,1,0,"DATI 1-100"
80 FOR I=1 TO 100
85 J=J+1
90 INPUT#2,D
95 IF J<> 8 THEN 115
100 PRINT"BATTI TASTO PER CONT."
105 GET CR$:IF CR$=""GOTO 105
110 J=0:PRINT"■"
115 PRINT:PRINT D,D↑2
120 NEXT I
125 CLOSE 2
130 END
```

Fig. 1 - Codifica del programma che in due fasi successive registra e rilegge da nastro magnetico una successione di dati numerici (per C64).

l'istruzione 55, la quale comunica appunto al computer che si è conclusa la fase di registrazione del file a cui è stato associato il numero 1 dell'istruzione **OPEN**.

L'interruzione temporanea dell'elaborazione avviene all'istruzione 6Ø, quando cioè è necessario riavvolgere il nastro all'inizio del file-dati per consentire alla successiva parte del programma di effettuare la "rilettura" del file chiamato "DATI 1-1ØØ". Quest'ultima parte del programma viene attivata battendo sulla tastiera il comando BASIC di continuazione:

CONT

con il quale l'elaborazione prosegue dall'istruzione successiva allo **STOP** temporaneo.

L'istruzione 7Ø riapre il file-dati "DATI 1-1ØØ", utilizzando il numero di buffer 2 per la lettura (Ø) delle informazioni registrate su nastro (1).

La linea di programma 9Ø provvede appunto a leggere i dati mediante l'istruzione **INPUT** #2, D, dove 2 è il numero corrispondente al file che è

stato riaperto e D è il valore della variabile letta (numeri da 1 a 1ØØ).

L'istruzione condizionale 1ØØ consente di stabilire una pausa per la visione sullo schermo dei risultati dell'elaborazione. Questa consiste nel calcolo e nella visualizzazione dei quadrati dei primi 1ØØ numeri interi (istruzione 115).

L'elaborazione prosegue con successivi calcoli e visualizzazioni di dieci gruppi di numeri per volta (istruzione 95).

Ad ogni intervallo l'operatore deve battere un **tasto qualsiasi della tastiera** per continuare l'elaborazione.

Come nel caso della scrittura, al termine della lettura dei dati il file viene chiuso con l'istruzione 125 (**CLOSE** 2).

Una variante dell'istruzione di lettura **INPUT #** è la **GET #**.

Questa istruzione consente l'immissione nella memoria principale di un "solo" carattere per volta dal nastro ed è molto usata quando si vogliono analizzare, carattere per carattere, i dati di un file-dati sequenziale.

Per esempio, l'istruzione

GET # 2, CR$

invia da nastro il primo carattere del file-dati 2 nella locazione di memoria CR$. Per primo carattere si intende il carattere posizionato immediatamente dopo la testina di lettura/scrittura dell'unità a nastro.

Come già precisato, nel programma in esame l'istruzione di arresto temporaneo (linea di programma 6Ø) consente di effettuare l'avvolgimento manuale (utilizzando il tasto REWIND dell'unità a nastro) del file-dati.

Questa operazione può essere evitata con i sistemi personal computer piú evoluti, che prevedono allo scopo l'utilizzo di apposite istruzioni di riavvolgimento del nastro magnetico o del posizionamento dello stesso su determinati file.

Se il file-dati fosse stato registrato su un floppy-disk, per il riposizionamento del file all'inizio sarebbe sufficiente rieseguire l'istruzione di apertura del file: **OPEN** # 1, "DATI 1-1ØØ".

Questa istruzione, infatti, non soltanto serve a definire i nomi dei file ed i numeri di buffer ad essi associati, ma anche a posizionare la testina di lettura/scrittura all'inizio del flusso d'informazioni.

Terminato l'esame del listato della *fig. 1*, può passare a batterlo sulla tastiera del computer.

Se utilizza il VIC-20, deve modificare le istruzioni 95 e 100, a causa delle minori dimensioni dello schermo, come di seguito indicato:

```
:
95 IF J <>8 THEN 115
100 PRINT "BATTI TASTO PER CONT."
:
```

Con il VIC-20 saranno quindi visualizzati al massimo otto gruppi di numeri per videata.

Dopo avere immesso il programma in questione nel computer, può accingersi ad eseguirlo.

Per compiere questa operazione deve, in primo luogo, disporre nell'apposito vano del registratore una cassetta a nastro su cui verrà registrato il file-dati, costituito dai primi 100 numeri interi, creata dal programma.

Può usare a questo scopo il lato B della terza cassetta a nastro, ricevuta con il 10° gruppo di lezioni, oppure qualsiasi cassetta tipo C30, purché sia di buona qualità.

Dopo avere introdotto la cassetta nell'apposita sede del registratore, riavvolga completamente il nastro magnetico all'inizio.

Terminate queste fasi di lavoro preliminari, batta il comando BASIC **RUN**.

Sullo schermo comparirà il messaggio:

PRESS RECORD & PLAY ON TAPE

Prema perciò contemporaneamente i tasti del registratore RECORD e PLAY.

Trascorsi alcuni secondi dal precedente messaggio, a questo si aggiungerà la seguente scritta:

```
OK
BREAK IN 60
READY
```

A questo punto il nastro magnetico si fermerà automaticamente ed il file, creato dal programma, risulterà cosí registrato sul nastro stesso.

Durante la fase di registrazione del file si può notare, usando il computer C64, che il listato del programma compare alcune volte sullo schermo per brevissimi intervalli di tempo.

Per poter "leggere" i dati contenuti nel file, occorre ora riavvolgere manualmente il nastro.

Abbassi in primo luogo il tasto STOP del registratore e riavvolga completamente il nastro, premendo il tasto REWIND: ad operazione ultimata prema nuovamente il tasto STOP.

Batta quindi il comando BASIC **CONT**.

Sullo schermo apparirà la scritta:

```
PRESS PLAY ON TAPE
```

Prema il tasto PLAY del registratore; sullo schermo appariranno in sequenza le scritte:

```
SEARCHING FOR DATI 1-100
FOUND  DATI 1-100
 1          1
 2          4
 3          9
 4          16
 5          25
 :          :
```

Ossia verranno visualizzati sullo schermo i numeri da 1 a 100 creati con la prima parte del programma (file-dati), ed i corrispondenti quadrati calcolati con la seconda parte del programma stesso.

Al termine di ogni videata di dati, per visualizzare i successivi numeri costituenti il file ed i relativi quadrati, è necessario battere un tasto qualsiasi.

2. – Esempio pratico di utilizzo di file-sequenziale

Il problema gestionale-amministrativo che ci accingiamo a risolvere in questo capitolo è un esempio di come si possa utilizzare un file-dati organizzato sequenzialmente su nastro magnetico.

Questo file-dati può servire a risolvere altri problemi gestionali simili sul computer, come il calcolo delle retribuzioni del personale di una piccola azienda, la gestione e l'interrogazione di piccoli archivi informativi (per esempio: clienti di un'azienda, fornitori, parti di magazzino, vendite mensili, costi e ricavi di una famiglia, elenchi anagrafici e telefonici, ecc.).

Una condizione alla quale devono soddisfare questi archivi per poter essere facilmente gestiti da un programma di elaborazione sul computer è che tutte le informazioni contenute in essi siano prelevabili o registrabili in modo esclusivamente sequenziale: l'elaborazione deve cioè essere tale da utilizzare "uno per volta" ed "uno dopo l'altro" i dati registrati sul file.

Il problema che vogliamo elaborare riguarda il rimborso mensile delle spese sostenute dal personale viaggiante di un'azienda.

Alcuni dipendenti di tale azienda, aventi mansioni commerciali, potrebbero avere sostenuto durante il mese lavorativo spese di viaggio o di altro

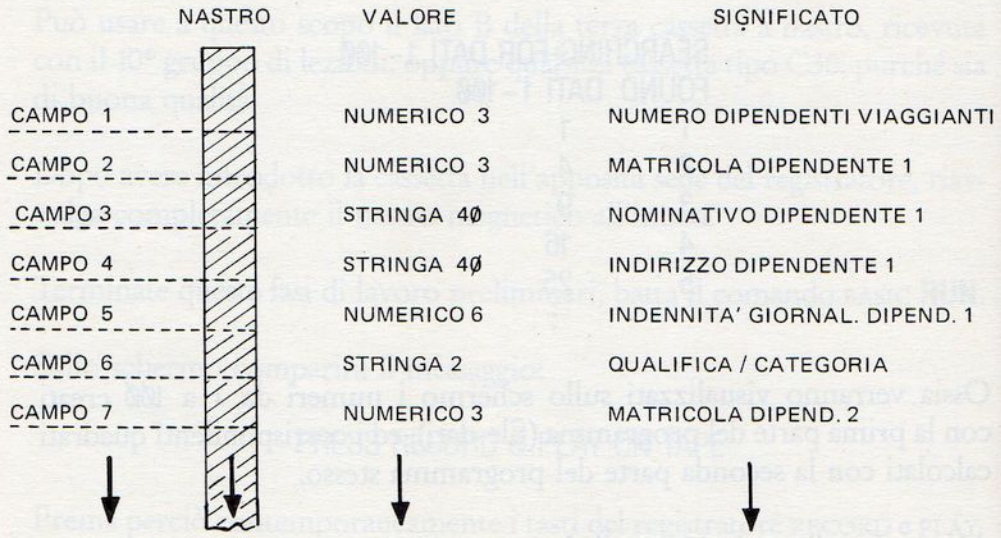


Fig. 2 – Informazioni di carattere generale dei dipendenti, organizzate in data-file su nastro magnetico.

genere e potrebbero avere ricevuto degli anticipi dalla ditta all'inizio dello stesso mese. Si tratta di valorizzare il totale delle spese sostenute da questi dipendenti, detratto degli eventuali anticipi ricevuti.

Ciò si ottiene moltiplicando una determinata tariffa giornaliera (che varia da categoria a categoria di dipendenti) per il numero dei giorni che un certo dipendente ha impiegato, per esempio, per visitare certi clienti della ditta.

Tutte le informazioni di carattere generale relative ai dipendenti possono essere organizzate e registrate in un data-file sequenziale (che chiameremo DATA-DIP nel programma) su nastro magnetico.

Nella *fig. 2* sono rappresentate schematicamente le informazioni di carattere generale dei dipendenti dell'azienda, organizzate in data-file su nastro.

Tutte le informazioni riguardanti invece il numero di giorni per cui si è protratto il viaggio di ciascun dipendente e gli eventuali anticipi ricevuti sono anch'esse archiviate sequenzialmente in istruzioni-archivio all'interno di un secondo programma, denominato NOTA SPESE, cioè sono elencate in istruzioni **DATA** ed immesse nella memoria principale in tre campi, M2, C, D, per mezzo dell'istruzione **READ**.

Nella *fig. 3* sono schematizzati i vari campi elencati nelle istruzioni **DATA** con i relativi significati.

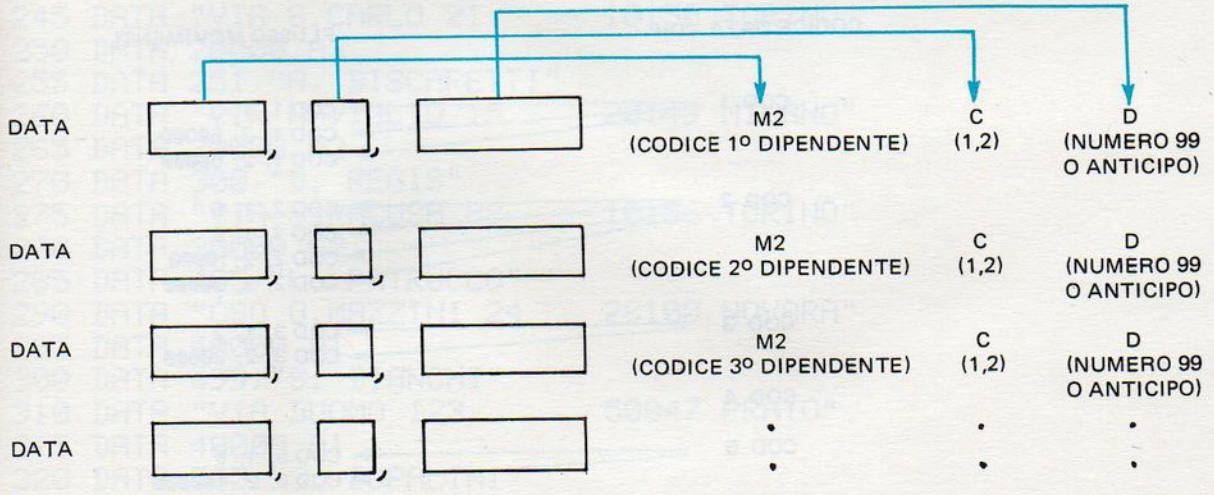


Fig. 3 – Campi elencati nelle istruzioni **DATA** *del programma "NOTA SPESE" con i relativi significati.*

Come si può notare dal prospetto di tale figura, il valore contenuto in M2 rappresenta il codice o matricola del generico impiegato (dipendente viaggiante). Questo codice ha la stessa struttura in caratteri del codice o matricola registrato nel file-archivio DATA-DIP precedente (tre caratteri numerici).

Il valore di C, invece, rappresenta un codice operativo che può essere 1 oppure 2.

Se C=1, il successivo valore D rappresenta un numero di giorni; se C=2, il valore di D rappresenta un anticipo in lire.

Tutto questo naturalmente è relativo al dipendente viaggiante.

In pratica, per tale dipendente vengono immessi nella memoria del computer la matricola corrispondente (M2), il numero dei giorni impegnati nel corso del viaggio per visitare i clienti (D=numero dei giorni nel caso in cui C=1), oppure un certo anticipo in lire (D=valore monetario nel caso in cui C=2).

In conclusione, il programma BASIC preleverà un dato per volta dai due flussi precedentemente descritti (il flusso DATA-DIP registrato su nastro e quello riportato in istruzioni **DATA**). Per ogni codice di dipendente che coincida con entrambi i due flussi sequenziali vengono effettuate la valorizzazione e totalizzazione delle spese giornaliere, nonché la detrazione degli eventuali anticipi.

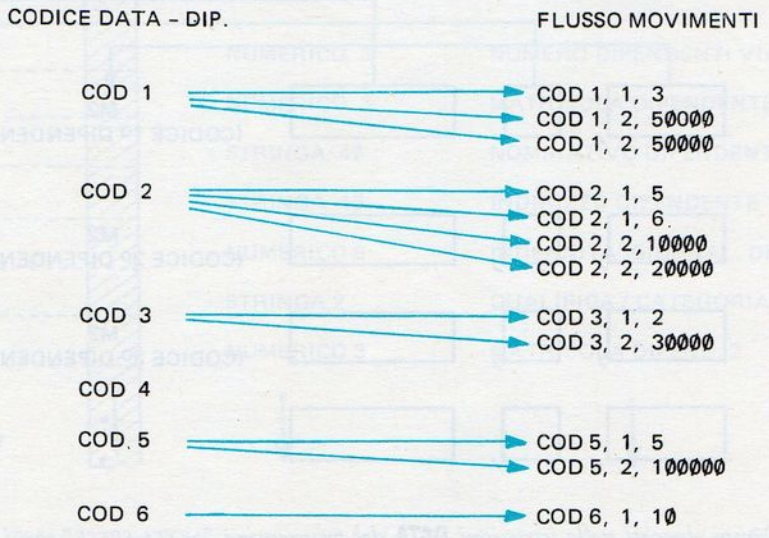


Fig. 4 – Flusso dati e flusso movimenti.

Relativamente ad una stessa matricola si potrebbero avere piú movimenti; in tal caso il programma provvederà a valorizzare ed a totalizzare questi movimenti (nel caso in cui C=1, verranno valorizzate le spese dei giorni di viaggio; nel caso in cui C=2, verranno detratti dalle spese sostenute gli anticipi ricevuti). L'elaborazione procede passo dopo passo per codice di dipendente; i due flussi sequenziali di dati sono organizzati in modo che sia possibile effettuare un "accoppiamento" tra essi.

```
10 REM*CREAZIONE FILE*
15 REM*   DATA-DIP   *
20 OPEN 1,1,1,"DATA-DIP"
25 READ C,NO$,IN$,D,Q$
30 IF C=1000 THEN 55
40 PRINT#1,C:PRINT#1,NO$:PRINT#1,IN$:PRINT#1,D:PRINT#1,Q$
45 PRINT C:PRINT NO$:PRINT IN$:PRINT D:PRINT Q$:GOTO 25
55 CLOSE 1:STOP
60 REM*AVVIARE IL NASTRO*
65 REM*   MANUALMENTE   *:PRINT"■"
70 OPEN 2,1,0,"DATA-DIP"
90 INPUT#2,C,NO$,IN$,D,Q$
93 IF C=999 THEN CLOSE 2:STOP
95 PRINT C:PRINT NO$:PRINT IN$:PRINT D :PRINT Q$
100 PRINT"BATTI UN TASTO":PRINT"PER CONTINUARE"
110 GET CR$:IF CR$=""GOTO 110
120 GOTO 90
210 DATA 150,"M. ANTONAZZI"
220 DATA "VIA DEL CORSO 3      10126 TORINO"
230 DATA 20000,A3
240 DATA 169,"G. ROSSI"
245 DATA "VIA S.CARLO 21       10150 TORINO"
250 DATA 20000,A3
255 DATA 251,"A. BISCARETTI"
260 DATA "VIA NAVIGLIO 15      20149 MILANO"
265 DATA 30000,A2
270 DATA 300,"D. REGIS"
275 DATA "VIA SIRACUSA 88      10156 TORINO"
280 DATA 30000,A2
285 DATA 405,"L. PATRUCCO"
290 DATA "CSO G.MAZZINI 24     28100 NOVARA"
295 DATA 40000,A1
300 DATA 499,"S. BIANCHI"
310 DATA "VIA DUOMO 123        50047 PRATO"
315 DATA 40000,A1
320 DATA 505,"S. BUFALINI"
325 DATA "PIAZZA CADORE 1      20100 MILANO"
```

Fig. 5 – Programma per la creazione del file-dati "DATA-DIP" di una ipotetica azienda commerciale.

```
330 DATA 20000,A3
335 DATA 560,"O. MATTEO"
340 DATA "VIA DELLE ROSE 9     20149 MILANO"
345 DATA 30000,A2
350 DATA 600,"M. GABRIELLI"
355 DATA "VIA MONTE ROSA 5     10130 TORINO"
360 DATA 40000,A1
365 DATA 608,"A. CECCONE"
370 DATA "VIA A.MANZONI 60     11100 PIANEZZA"
375 DATA 30000,A2
380 DATA 999,"*"
385 DATA "********"
390 DATA 1000,A2
400 DATA 1000,"*"
410 DATA "********"
420 DATA 1000,A2
```

Fig. 6 – Proseguimento del programma per la creazione del file-dati "DATA-DIP".

In altre parole, ad un codice letto dal primo flusso (DATA-DIP) devono corrispondere uno o piú movimenti relativi allo stesso codice, come schematizzato nel prospetto riportato nella *fig. 4*. In tale esempio, come si può notare, la matricola con codice 4 non ha effettuato movimenti.

Il flusso movimenti viene anche definito **flusso-guida**, nel senso che da esso partono le richieste di informazioni di carattere generale, da ricercare sull'altro flusso di dati (DATA-DIP).

La creazione e la registrazione dei file-dati relativi a tutto il personale viaggiante dell'azienda vengono effettuate con l'apposito programma BASIC riportato nella *fig. 5* e nella *fig. 6*.

In questo programma sono stati utilizzati, a titolo esemplificativo, dei nominativi del tutto generici (istruzioni DATA 21Ø÷375). E' possibile comunque estendere l'archivio con l'aggiunta di altre istruzioni **DATA** dopo il numero 375.

Ogni gruppo di tre **DATA** contiene le informazioni generali relative ad ogni dipendente viaggiante; di queste informazioni la prima, che è un numero intero di tre cifre, riguarda il codice o matricola del dipendente.

Ad esempio, nell'istruzione 21Ø il codice del primo dipendente è 15Ø; nell'istruzione 24Ø il codice del secondo dipendente è 169 e cosí via di seguito.

I codici, come si vede, sono disposti in ordine crescente, nello stesso ordine con cui saranno disposti i codici dei dipendenti nelle istruzioni del programma successivo, denominato NOTA SPESE, il quale provvederà a valorizzare le effettive spese sostenute dai dipendenti viaggianti.

Nel programma della *fig. 5* e della *fig. 6*, l'istruzione 2Ø provvede ad aprire il file-dati per la registrazione sul nastro dei dati corrispondenti a ciascun dipendente viaggiante.

Nell'istruzione 25 le variabili di nome C, NO$, IN$, D, Q$ vengono associate ai corrispondenti dati riportati nelle istruzioni **DATA** 21Ø÷375, i quali vengono poi visualizzati con le istruzioni **PRINT** presenti nella linea di programma 45.

In particolare, tali istruzioni provvedono a stampare il numero di codice o matricola (C), le stringhe NO$ (nome e cognome), IN$ (indirizzo), la cifra corrispondente all'indennità giornaliera (D) ed infine la qualifica o categoria (Q$).

L'istruzione 4Ø ha il compito di registrare sequenzialmente sul nastro i dati di ciascun dipendente viaggiante, i quali costituiscono il file-dati. Al termine della scrittura di questi dati, il file viene chiuso (linea di programma 55).

L'istruzione **STOP**, presente nella stessa linea di programma 55, ha la funzione di consentire il riavvolgimento manuale del nastro magnetico al termine dell'operazione di registrazione, predisponendolo per la lettura.

A questo scopo l'istruzione 7Ø, utilizzando il numero 2 di buffer, provvede a riaprire il file "DATA-DIP" per la lettura dei dati registrati. La linea di programma 9Ø consente appunto di leggere i dati costituenti il file.

Dopo l'istruzione 9Ø il programma si chiede (riga 93) se C=999: in caso affermativo, il file 2 viene chiuso. Ciò significa che la lettura del file-archivio "DATA-DIP" è terminata.

L'istruzione 11Ø consente, battendo un tasto qualsiasi, di visualizzare sullo schermo, uno per volta, tutti i dati di ciascun dipendente.

L'analisi del programma è cosí terminata. **Prima di eseguirlo è però necessario esaminare il programma NOTA SPESE, il quale, come già accennato, verrà utilizzato con quello ora descritto.**

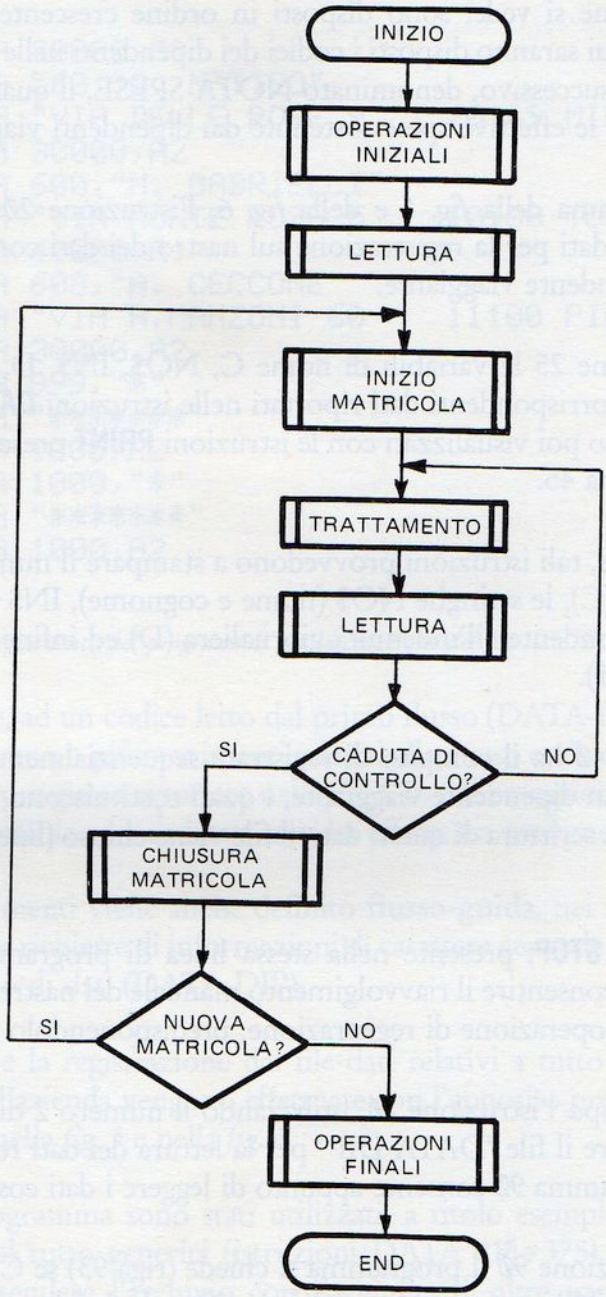


Fig. 7 – Diagramma di flusso del programma "NOTA-SPESE".

Nella *fig. 7* è riportato il diagramma di flusso appunto del programma NOTA SPESE, mentre nella *fig. 8* e nella *fig. 9* sono rappresentate piú in dettaglio alcune subroutine dello stesso diagramma.

Come può osservare, il programma in questione è costruito in modo modulare, ossia è costituito da diversi moduli che operano a guisa di sottoprogrammi.

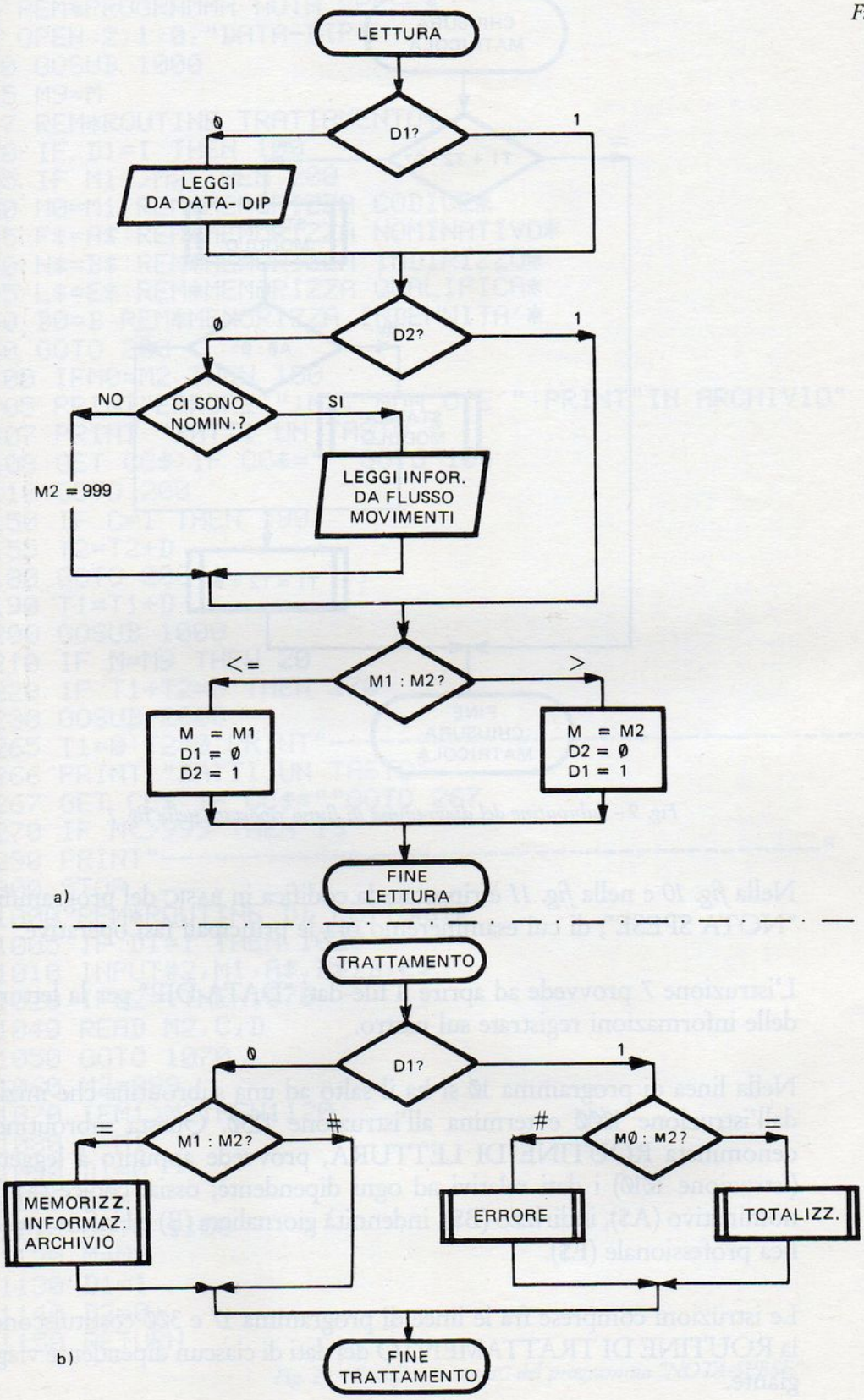


Fig. 8 – Subroutine del diagramma di flusso riportato nella fig. 7.

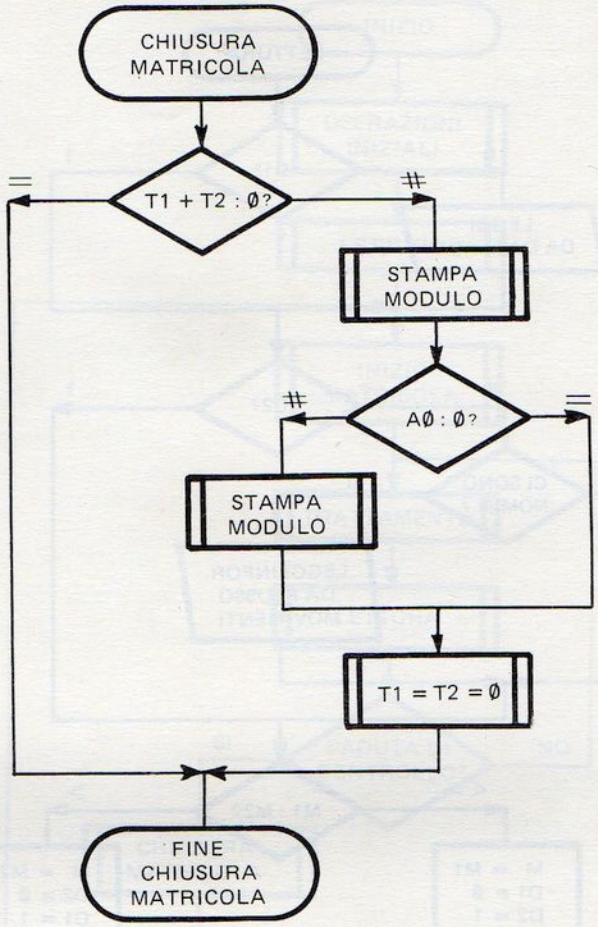


Fig. 9 – Subroutine del diagramma di flusso riportato nella fig. 7.

Nella *fig. 10* e nella *fig. 11* è riportata la codifica in BASIC del programma "NOTA SPESE", di cui esamineremo ora le principali fasi operative.

L'istruzione 7 provvede ad aprire il file-dati "DATA-DIP" per la lettura delle informazioni registrate sul nastro.

Nella linea di programma 1Ø si ha il salto ad una subroutine che inizia dall'istruzione 1ØØØ e termina all'istruzione 115Ø. Questa subroutine, denominata ROUTINE DI LETTURA, provvede appunto a leggere (istruzione 1Ø1Ø) i dati relativi ad ogni dipendente, ossia: codice (M1), nominativo (A$), indirizzo (B$), indennità giornaliera (B) ed infine qualifica professionale (E$).

Le istruzioni comprese fra le linee di programma 17 e 3ØØ costituiscono la ROUTINE DI TRATTAMENTO dei dati di ciascun dipendente viaggiante.

```
5 REM*PROGRAMMA NOTA SPESE*
7 OPEN 2,1,0,"DATA-DIP"
10 GOSUB 1000
15 M9=M
17 REM*ROUTINE TRATTAMENTO*
20 IF D1=1 THEN 100
25 IF M1<>M2 THEN 200
30 M0=M1:REM*MEMORIZZA CODICE*
35 F$=A$:REM*MEMORIZZA NOMINATIVO*
40 H$=B$:REM*MEMORIZZA INDIRIZZO*
45 L$=E$:REM*MEMORIZZA QUALIFICA*
50 B0=B:REM*MEMORIZZA INDENNITA'*
60 GOTO 200
100 IFM0=M2 THEN 150
105 PRINT"ERRORE:";M2;"NON C'E'":PRINT"IN ARCHIVIO"
107 PRINT "BATTI UN TASTO "
108 GET CC$:IF CC$="" GOTO 108
110 GOTO 200
150 IF C=1 THEN 190
155 T2=T2+D
180 GOTO 200
190 T1=T1+D
200 GOSUB 1000
210 IF M=M9 THEN 20
220 IF T1+T2=0 THEN 270
230 GOSUB 2000
265 T1=0:T2=0:PRINT"-----------------------------------------------":PRINT
266 PRINT "BATTI UN TASTO"
267 GET CC$:IF CC$=""GOTO 267
270 IF M<>999 THEN 15
290 PRINT"-----------------------------------------------"
300 STOP
1000 REM*ROUTINE DI LETTURA*
1005 IF D1=1 THEN 1020
1010 INPUT#2,M1,A$,B$,B,E$
1020 IFD2=1THEN1070
1040 READ M2,C,D
1050 GOTO 1070
1060 M2=999
1070 IFM1>M2THEN1120
1080 M=M1
1090 D1=0
1100 D2=1
1110 GOTO 1150
1120 M=M2
1130 D1=1
1140 D2=0
1150 RETURN
```

Fig. 10 – Codifica in BASIC del programma "NOTA-SPESE".

```
2000 REM*ROUTINE CHIUSURA MATRICOLA*
2010 PRINT"□"
2015 PRINT"----------------------------------------"
2020 PRINTF$
2025 PRINT"MATR./CAT.";M0;"/";L$
2030 PRINTH$
2035 PRINT"INDENNITA'/GIOR.";B0
2040 PRINT"PER";T1;"GIORNI"
2045 PRINT
2050 PRINT"TOT.RIMB.=";B0*T1
2055 PRINT"TOT.ACC.=";T2
2060 W=T2-T1*B0
2070 IF W<0 THEN 2320
2300 PRINT"DIFF.A DEBITO=";W
2310 GOTO2330
2320 PRINT"DIFF.A CREDITO=";-W
2330 RETURN
3000 DATA 251,1,3
3010 DATA 251,1,2
3020 DATA 251,2,35000
3030 DATA 251,2,25000
3060 DATA 380,1,7
3070 DATA 380,2,80000
3080 DATA 499,1,6
3090 DATA 499,1,4
3100 DATA 499,1,3
3110 DATA 832,1,5
3120 DATA 842,1,7
3130 DATA 842,1,2
3140 DATA 842,2,60000
3150 DATA 842,2,40000
3160 DATA 842,2,55000
3170 DATA 870,1,5
```

Fig. 11 – Proseguimento del listato programma "NOTA-SPESE".

In particolare, l'istruzione 1Ø5 provvede a visualizzare sullo schermo un messaggio che avverte l'operatore quando il codice di un determinato dipendente non è presente in archivio.

Alla linea di programma 23Ø si ha il richiamo ad un'altra subroutine, denominata ROUTINE CHIUSURA MATRICOLA (istruzione 2ØØØ), che provvede ad eseguire il calcolo delle spese effettuate dai dipendenti viaggianti.

Per ciascun dipendente "movimentato" (cioè che ha effettuato viaggi) si

ha sullo schermo la visualizzazione di un prospetto in cui appaiono, oltre che le generalità dell'interessato (nominativo, indirizzo dell'abitazione, codice, ecc.), i valori delle spese effettuate.

Il flusso sequenziale dei movimenti è riportato, come si vede, nelle istruzioni DATA 3ØØØ÷317Ø. E' questo il flusso che deve essere "accoppiato" con quello del file "DATA-DIP", registrato sul nastro magnetico.

Per rendere operativa l'intera procedura, costituita dai due programmi "CREAZIONE DATA-DIP", "NOTA SPESE" e dal file-dati sequenziale "DATA-DIP", occorre che quest'ultimo sia registrato sul nastro subito dopo il programma "NOTA SPESE".

Non è necessario invece che il programma "CREAZIONE DATA-DIP" si trovi sullo stesso nastro su cui sono registrati gli altri due, ma è sufficiente che il file-archivio "DATA-DIP" sia disponibile per la lettura sequenziale subito dopo il programma principale "NOTA SPESE".

Per eseguire l'intera procedura è consigliabile utilizzare una cassetta vergine tipo C30, facilmente reperibile in commercio.

Non disponendo di una cassetta vergine, può utilizzare per questo scopo una cassetta audio in Suo possesso già registrata, purché in buono stato, che ovviamente dovrà prima cancellare.

Per eseguire la cancellazione deve fare scorrere il nastro magnetico mantenendo premuti i tasti RECORD e PLAY del registratore. Tale operazione deve essere effettuata su entrambi i lati della cassetta.

Su un lato della cassetta vergine (*fig. 12*), come troverà specificato piú oltre, dovrà registrare i programmi "NOTA SPESE" e "CREAZ. DATA-DIP" e sul lato opposto l'archivio "DATA-DIP".

A questo punto può passare a battere sulla tastiera il programma "NOTA SPESE" (*figg. 10* e *11*).

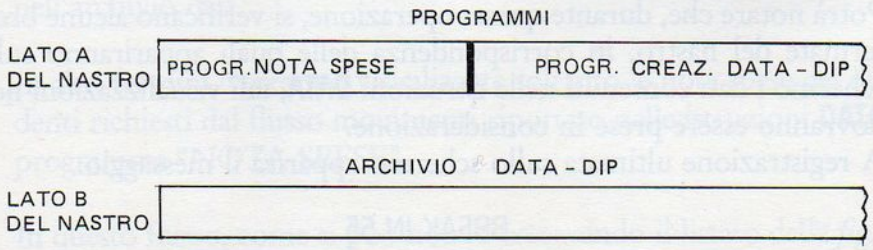


Fig. 12 – Registrazione dei programmi sulla cassetta.

Se utilizza il computer VIC-20, dovrà ovviamente modificare le istruzioni 265, 29Ø e 2Ø15, battendo 21 trattini orizzontali anziché 39.

Al termine della battitura è opportuno **controllare attentamente il listato che appare sullo schermo** con quello riportato nella *fig. 10* e nella *fig. 11*, correggendo gli eventuali errori.

Per il momento il programma in questione non deve essere eseguito.

- Registri ora il programma "NOTA SPESE" su uno dei due lati (che chiameremo A) della cassetta vergine (*fig. 12*).
- Cancelli quindi la memoria del computer con il comando BASIC **NEW.**
- Batta quindi sulla tastiera del computer il programma "CREAZIONE FILE DATA-DIP" riportato nella *fig. 5* e nella *fig. 6*.
- Terminata la battitura controlli, come al solito, il listato che appare sullo schermo, quindi registri il programma in questione (che potrà chiamare DATA-DIP) subito dopo il programma "NOTA SPESE" precedentemente registrato.
- Al termine dell'operazione estragga la cassetta dal relativo vano del registratore e la rimetta al suo posto in modo che l'altro lato, che chiameremo B, sia rivolto verso l'alto. Questo lato del nastro costituisce l'archivio dei dati.
- Riavvolga quindi completamente il nastro all'inizio per poter procedere alla registrazione del "DATA-FILE".
- Esegua ora il programma attualmente presente nella memoria del computer, ossia il programma "CREAZIONE FILE DATA-DIP", battendo il comando **RUN**.
  Sullo schermo verrà visualizzato il messaggio:

  PRESS RECORD & PLAY ON TAPE

  Premendo questi due tasti del registratore, il nastro verrà posto in movimento per la registrazione dei dati.
  Potrà notare che, durante questa operazione, si verificano alcune brevi fermate del nastro, in corrispondenza delle quali appariranno sullo schermo i dati contenuti nelle istruzioni **DATA**; tali visualizzazioni non dovranno essere prese in considerazione.
  A registrazione ultimata sullo schermo apparirà il messaggio:

  BREAK IN 55

- A questo punto estragga la cassetta e la disponga nuovamente nella sua sede sul registratore, con il lato A rivolto verso l'alto.

- Riavvolga completamente il nastro e carichi nella memoria del computer il programma "NOTA SPESE", battendo il comando **LOAD**.

- A caricamento avvenuto, estragga la cassetta e la disponga con il lato B rivolto verso l'alto.

- Riavvolga completamente il nastro. A questo punto tutto è pronto per l'esecuzione del programma "NOTA SPESE".

- Batta il comando **RUN**.
  Sullo schermo apparirà la scritta:

```
PRESS PLAY ON TAPE
```

Dopo avere premuto il tasto PLAY del registratore, sullo schermo verranno visualizzate le note spese del dipendente il cui codice (251) è riportato nella prima istruzione **DATA** del programma "NOTA SPESE", ossia:

```
A. BISCARETTI
MATR./CAT. 251/A2
VIA NAVIGLIO 15        20149 MILANO
INDENNITA'/GIOR. 30000
PER 5 GIORNI
TOT. RIMB.       =150000
TOT. ACC.        = 60000
DIFF. A CREDITO  = 90000
```

Battendo un tasto qualsiasi della tastiera, sullo schermo comparirà la scritta:

```
ERRORE 380 NON C'E'
IN ARCHIVIO
```

la quale indica appunto che il codice di tale dipendente non è presente nell'archivio dati.

In altri termini, verranno visualizzate soltanto le note spese dei dipendenti richiesti dal flusso movimenti riportato nelle istruzioni **DATA** del programma "NOTA SPESE".

In questo flusso, come si può notare osservando il listato della *fig. 10* e

della *fig. 11*, sono riportati anche movimenti di dipendenti inesistenti, i cui codici cioè non sono presenti nell'archivio "DATA-DIP" (ad esempio, i codici dipendenti 38Ø, 832, 842, ecc.).

In particolare, per ciascun movimento di dipendente inesistente (corrispondente ad una **DATA**) verrà visualizzato il messaggio che ne indica l'assenza in archivio.

Per esempio, in corrispondenza del codice 38Ø comparirà sullo schermo due volte il messaggio prima citato, il quale verrà riprodotto sei volte in corrispondenza del codice 842, ecc.

Al termine dell'esecuzione del programma, ossia dopo la visualizzazione della scritta indicante la mancanza in archivio del codice dipendente 87Ø, apparirà il messaggio:

```
?OUT OF DATA ERROR IN 1040
```

il quale indica appunto che è stata tentata, senza esito, la lettura di un dato dopo il numero di codice 87Ø riportato nell'ultima istruzione **DATA** del programma della *fig. 11*.

3. - Operatività dei tasti di funzione

Sul lato destro dei computer C64 e VIC-20 sono presenti quattro tasti, contraddistinti dalle scritte f1 | f2 , f3 | f4 , f5 | f6 , f7 | f8 , i quali, come già precisato nella *Pratica 1*, sono indicati genericamente con il termine TASTI DI FUNZIONE.

Pigiando tali tasti, sullo schermo non succede apparentemente nulla.

In realtà, ciascun tasto genera, all'interno del computer, un corrispondente codice ASCII, che può essere utilizzato in diversi modi se è abbinato ad una specifica operazione prevista in un programma.

Ad esempio, in un programma si può prevedere di avere, premendo un tasto di funzione, la stampa di una stringa di caratteri oppure il salto ad una subroutine.

Quanto ora precisato può essere ottenuto anche pigiando un qualsiasi

TASTI DI FUNZIONE	C64		VIC 20	
	SIMBOLI (MAIUSCOLO)	SIMBOLI (MINUSCOLO)	SIMBOLI (MAIUSCOLO)	SIMBOLI (MINUSCOLO)
f1		E		E
f2 = SHIFT → f1		I		I
f3		F		F
f4 = SHIFT → f3		J		J
f5		G		G
f6 = SHIFT → f5		K		K
f7		H		H
f8 = SHIFT → f7		L		L

Fig. 13 – Raffigurazione grafica ottenuta sullo schermo del C64 e del VIC-20 quando vengono impostati i tasti di funzione all'interno di una stringa.

altro tasto della tastiera, però l'uso dei tasti funzionali, essendo questi separati dagli altri, risulta piú pratico. Inoltre, i tasti di funzione, potendo eseguire soltanto le operazioni previste dal programmatore, consentono di eliminare qualsiasi possibilità d'errore.

Come già accennato in precedenza, le funzioni f1-f3-f5-f7 vengono attivate pigiando direttamente i corrispondenti tasti, mentre le funzioni f2-f4-f6-f8 vengono attivate premendo contemporaneamente il tasto **SHIFT** .

I tasti di funzione, se impostati all'interno di una stringa, ossia dopo avere battuto i caratteri " (virgolette), fanno comparire sullo schermo simboli stenografici che rappresentano le funzioni stesse, come si vede nella *fig. 13*.

Nelle colonne 2 e 4 di tale figura sono riportati rispettivamente i simboli che appaiono sullo schermo del C64 e del VIC-20 quando i computer sono disposti nel modo (SET) maiuscolo/grafico, ossia nelle normali condizioni di funzionamento.

Quando invece i computer sono disposti nel modo minuscolo (tasti C= → **SHIFT** premuti contemporaneamente), i simboli che compaiono

sullo schermo sono lettere dell'alfabeto su sfondo invertito (colonne 3 e 5).

Può verificare praticamente quanto sinora precisato battendo il carattere " (virgolette) e premendo uno per volta i tasti di funzione, prima con il computer nelle normali condizioni di funzionamento (modo maiuscolo) e poi nel modo minuscolo.

Nella tabella della *fig. 14* sono riportati i codici ASCII corrispondenti ai tasti di funzione.

TASTI	CODICE ASCII
f1	133
f2 = SHIFT → f1	137
f3	134
f4 = SHIFT → f3	138
f5	135
f6 = SHIFT → f5	139
f7	136
f8 = SHIFT → f7	140

Fig. 14 – Codici ASCII dei tasti di funzione.

Battendo per esempio la seguente istruzione ad esecuzione immediata:

PRINT ASC (" f1 ")

sullo schermo comparirà il numero:

133

relativo appunto al codice ASCII del tasto f1 .

Battendo l'istruzione:

```
10 REM*PROGRAMMA PROVA TASTI*
15 REM*     DI FUNZIONE      *
20 PRINT"■"
22 C=53281
25 GET A$:IF A$=""THEN 25
30 IF A$="■" THEN POKE C,0
35 IF A$="■" THEN POKE C,1
40 IF A$="■" THEN POKE C,2
45 IF A$="■" THEN POKE C,3
50 IF A$="■" THEN POKE C,4
55 IF A$="■" THEN POKE C,5
60 IF A$="■" THEN POKE C,6
65 IF A$="■" THEN POKE C,7
70 GOTO25
```

Fig. 15 - Listato del programma relativo alla prova dei tasti di funzione (per C64).

PRINT ASC (" SHIFT → f1 ")

comparirà il valore numerico:

137

corrispondente al codice ASCII del tasto f2, e cosí via di seguito.

Per potere utilizzare i codici dei tasti di funzione è necessario prevedere nel programma una routine che rilevi la pressione di un tasto.

Solamente nel caso in cui si prema il tasto prestabilito, verrà effettuata una data operazione nel programma stesso.

Verificheremo ora, praticamente, l'operatività dei tasti di funzione eseguendo un semplice programma che consente di ottenere il cambiamento dei colori dello schermo mediante la pressione appunto dei tasti di funzione.

Il listato del programma in questione per il C64 è riportato nella *fig. 15*, mentre nella *fig. 16* è rappresentato il listato adatto per il VIC-20.

I due listati differiscono unicamente per i diversi valori delle locazioni di memoria che controllano il colore dello schermo o sfondo (istruzioni 30÷65).

L'istruzione 22 di tali listati provvede ad assegnare alla variabile C la

```
10 REM*PROGR. PROVA*
12 REM*  TASTI DI  *
15 REM*  FUNZIONE  *
20 PRINT"■"
22 C=36879
25 GET A$:IF A$=""THEN 25
30 IF A$="■" THEN POKE C,11
35 IF A$="■" THEN POKE C,27
40 IF A$="■" THEN POKE C,43
45 IF A$="■" THEN POKE C,59
50 IF A$="■" THEN POKE C,75
55 IF A$="■" THEN POKE C,91
60 IF A$="■" THEN POKE C,107
65 IF A$="■" THEN POKE C,123
70 GOTO 25
```

Fig. 16 – Listato del programma relativo alla prova dei tasti di funzione (per VIC-20).

locazione di memoria relativa al colore dello schermo (53281 per il C64 e 36879 per il VIC-20).

La linea di programma 25 ha la funzione di attendere la pressione di un tasto.

Premendo uno per volta i tasti di funzione (istruzioni 30÷65), si otterrà il cambiamento del colore dello schermo.

Piú precisamente, nell'istruzione 30 è prevista la pressione del tasto f1 per ottenere lo schermo di colore nero (**POKE** C, 0 per C64 e **POKE** C, 11 per VIC-20); nella linea di programma 35 è prevista la pressione del tasto f2 per ottenere lo schermo di colore bianco (**POKE** C, 1 per C64 e **POKE** C, 27 per VIC-20) e cosí via.

Batta ora il listato del programma descritto e lo esegua con il comando **RUN**.

Premendo uno per volta i tasti di funzione, otterrà, come previsto, otto diversi colori dello schermo.

Per terminare il programma dovrà battere i tasti RUN | STOP e RESTORE.

Nella *fig. 17* è riportato il listato di un secondo programma che consente di verificare l'uso dei tasti di funzione.

```
10 REM*PROGRAMMA PROVA DEI TASTI*
15 REM*      DI FUNZIONE N. 2     *
20 PRINT"■"
25 GET A$:IF A$=""THEN 25
30 IF ASC(A$)=133 THEN PRINT CHR$(144):PRINT"■    ■"
35 IF ASC(A$)=137 THEN PRINT CHR$(5):PRINT"■      ■"
40 IF ASC(A$)=134 THEN PRINT CHR$(28):PRINT"■        ■"
45 IF ASC(A$)=138 THEN PRINT CHR$(159):PRINT"■          ■"
50 IF ASC(A$)=135 THEN PRINT CHR$(30):PRINT"■            ■"
55 IF ASC(A$)=139 THEN PRINT CHR$(158):PRINT"■              ■"
60 IF ASC(A$)=136 THEN PRINT CHR$(156):PRINT"■                ■"
65 GOTO 25
```

Fig. 17 – Listato di un altro programma relativo alla prova dei tasti di funzione (per C64).

In questo nuovo programma, battendo uno per volta i primi sette tasti di funzione, si otterrà la visualizzazione sullo schermo di sette strisce o barre colorate di diversa lunghezza (istruzione 3Ø÷6Ø).

Piú esattamente, nella linea di programma 3Ø viene confrontato il codice ASCII, relativo al carattere letto nella variabile A$, con il codice ASCII (133) corrispondente al tasto f1 . Se effettivamente viene premuto il tasto f1 , si ha la visualizzazione sullo schermo di una striscia di colore nero (CHR$ 144).

Analogamente, nella linea di programma 35 viene eseguito il confronto del codice ASCII del carattere letto nella variabile A$ con il codice ASCII (137) relativo al tasto f2 .

Se tale confronto è positivo, si ha la visualizzazione sullo schermo di una striscia di colore bianco (CHR$ 5), e cosí via di seguito.

In questo modo nelle linee di programma 4Ø, 45, 5Ø, 55 e 6Ø vengono eseguiti i confronti dei codici ASCII dei colori con i codici 134, 138, 135, 139 e 136, relativi ai tasti f3 , f4 , f5 , f6 e f7 , premendo i quali si otterranno corrispondentemente le strisce colorate in rosso (CHR$ 28), cyan (CHR$ 159), verde (CHR$ 3Ø), giallo (CHR$ 158) e porpora (CHR$ 156).

Cancelli ora dalla memoria del computer il precedente programma ed immetta in esso il listato della *fig. 17.*

Se si impiega il computer VIC-20, occorre modificare le istruzioni 1Ø,

15, 35 e aggiungere l'istruzione 12 come di seguito indicato.

```
10 REM*PROGR. PROVA*
12 REM*  TASTI DI  *
15 REM*FUNZIONE N.2*
20 .....
25 .....
30 .....
35 IF ASC(A$)=137 THEN PRINT CHR$(31):PRINT"■                ■"
40 .....
```

In questo caso la linea di programma 35 consente, pigiando il tasto **f2**, di visualizzare sullo schermo una striscia di colore blu (CHR$ 31).

Esegua, infine, il programma battendo il comando BASIC **RUN**.

Premendo i tasti f1÷f7, verranno visualizzate sullo schermo sette strisce di diverso colore e lunghezza.

A conclusione dell'argomento, eseguiamo ora un programma nel quale è sufficiente battere il tasto di funzione **f1** oppure il tasto **f3** per visualizzare sullo schermo rispettivamente il quadrato e la radice quadrata di un numero.

Nella *fig. 18* è riportato il listato di tale programma per il computer C64, mentre nella *fig. 19* è rappresentato il corrispondente listato per il VIC-20.

```
10 REM*  QUADRATO E RADICE  *
15 REM*QUADRATA DI UN NUMERO*
20 PRINT"■"
25 INPUT"■INTRODURRE IL NUMERO";N
30 PRINT"■BATTERE IL TASTO F1"
35 PRINT"PER CALCOLARE IL QUADRATO"
40 PRINT"■BATTERE IL TASTO F3"
45 PRINT"PER CALCOLARE LA RADICE QUADRATA"
50 PRINT"■BATTERE IL TASTO F5"
55 PRINT"PER FINIRE"
60 PRINT"----------------------------------------"
65 GET A$:IF A$=""THEN 65
70 IF A$="■" THEN PRINT"IL QUADRATO DI";N;"E'";N↑2
75 IF A$="■" THEN PRINT"LA RADICE QUADRATA DI";N;"E'";SQR(N)
80 IF A$="■" THEN END
100 FOR T=1 TO 3000:NEXT T:GOTO 20
```

Fig. 18 – Listato del programma che prevede l'uso dei tasti di funzione (per C64).

```
10 REM*QUADRATO E RA-*
12 REM*DICE QUADRATA *
15 REM*DI UN NUMERO  *
20 PRINT"■"
25 PRINT"■INTRODURRE IL NUMERO"
27 INPUT N
30 PRINT"■BATTERE IL TASTO F1"
35 PRINT"PER CALCOLARE IL"
37 PRINT"QUADRATO"
40 PRINT"■BATTERE IL TASTO F3"
45 PRINT"PER CALCOLARE LA"
47 PRINT"RADICE QUADRATA"
50 PRINT"■BATTERE IL TASTO F5"
55 PRINT"PER FINIRE"
60 PRINT"----------------------"
65 GET A$:IF A$=""THEN 65
70 IF A$="■"THEN PRINT"IL QUADRATO DI";N;"E'";N↑2
75 IF A$="■"THEN PRINT"LA RADICE QUADRATA DI";N;"E'";SQR(N)
80 IF A$="■"THEN END
100 FOR T=1 TO 3000:NEXT T:GOTO 20
```

Fig. 19 – Listato del programma che prevede l'uso dei tasti di funzione (per VIC-20).

La linea di programma 65 ha la funzione di attendere, come già visto, la pressione di un tasto.

All'istruzione 70 il programma si chiede se il tasto premuto è corrispondente a **f1** : in caso affermativo viene visualizzato il quadrato del numero N immesso nel computer tramite l'istruzione **INPUT**.

Analogamente, all'istruzione 75, se viene premuto il tasto **f3** , il programma effettua il calcolo della radice quadrata del numero N.

Infine, alla linea 80, se viene premuto il tasto **f5** , si ha la fine del programma.

Alla riga 100 vi è una breve pausa d'attesa, al termine della quale l'elaborazione ritorna all'istruzione 20, da cui il programma riprende ad elaborare.

Dopo avere cancellato il precedente programma, batta il listato contenuto nella *fig. 18* oppure nella *fig. 19* a seconda del computer in Suo possesso, e lo esegua con il comando **RUN**.

Sullo schermo comparirà il messaggio:

INTRODURRE IL NUMERO?

A questo punto può introdurre nel computer il numero di cui desidera effettuare il calcolo del quadrato oppure della radice quadrata.

Supponendo ad esempio di voler calcolare il quadrato di 5, dovrà battere tale numero.

Sullo schermo appariranno le seguenti scritte:

BATTERE IL TASTO F1
PER CALCOLARE IL QUADRATO
BATTERE IL TASTO F3
PER CALCOLARE LA RADICE QUADRATA
BATTERE IL TASTO F5
PER FINIRE

Se batterà il tasto f1, comparirà la scritta:

IL QUADRATO DI 5 E' 25

Dopo alcuni secondi tale scritta sarà cancellata dallo schermo e verrà nuovamente visualizzato il messaggio che invita l'operatore ad introdurre il numero N da calcolare.

Potrà completare l'esecuzione del programma effettuando il calcolo della radice quadrata di un qualsiasi numero N da Lei immesso nel computer, battendo il tasto f3.

4. - Mappe di memoria dei computer C64 e VIC-20

Allo scopo di approfondire le conoscenze strutturali del C64 e del VIC-20, esaminiamo ora, sinteticamente, come sono organizzate le diverse aree di memoria (RAM e ROM) di questi due computer.

Come già precisato in una precedente lezione, la memoria del calcolatore può essere configurata come una cassettiera, costituita da una serie di cassetti dotati di un indirizzo di riconoscimento (che va da Ø a 65535 in decimale e da ØØØØ a FFFF in esadecimale), ognuno dei quali può contenere una determinata quantità di dati.

COMPUTER C64

Nella *fig. 20* è riportata la mappa di memoria del C64. In essa le varie locazioni di memoria sono contraddistinte con numerazione esadecimale e decimale.

In tale tipo di memoria non vi è una netta distinzione fra le aree RAM e quelle ROM, per cui alcune sue parti possono appartenere indifferentemente all'area RAM oppure all'area ROM.

Vediamo ora quali sono le principali funzioni svolte dai diversi blocchi di memoria.

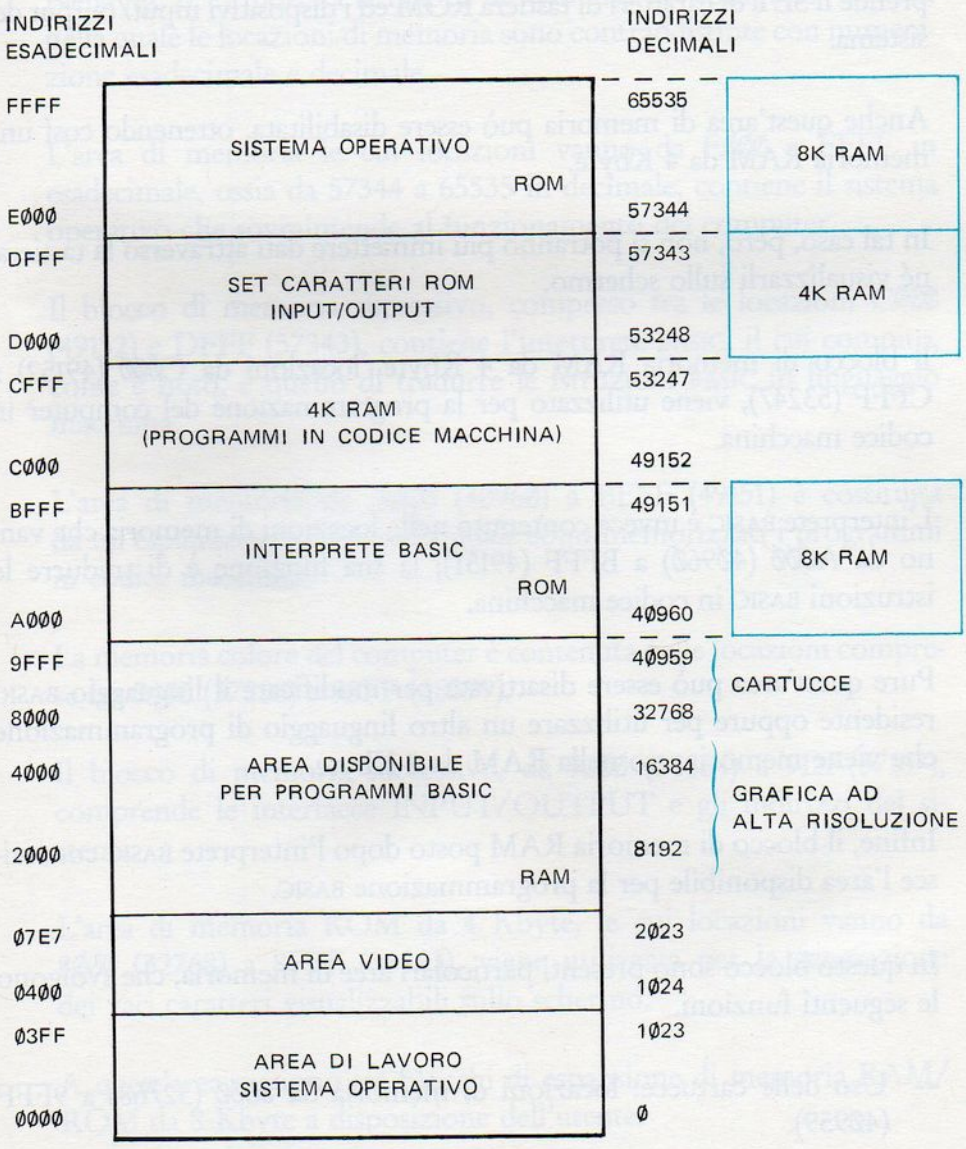


Fig. 20 – Mappa di memoria del C64.

L'area di memoria comprendente gli indirizzi da EØØØ a FFFF in esadecimale, ossia da 57344 a 65535 in decimale, contiene il sistema operativo del C64, ossia le istruzioni fondamentali per il funzionamento del computer.

Quest'area può essere disabilitata in modo da ottenere un'area di memoria RAM di 8 Kbyte.

In queste condizioni il computer non potrà piú rispondere, ovviamente, ai comandi immessi tramite la tastiera, per cui occorrerà adottare un altro sistema operativo.

La successiva area di memoria, da DØØØ (53248) a DFFF (57343), comprende il SET di caratteri di tastiera ROM ed i dispositivi input/output del sistema.

Anche quest'area di memoria può essere disabilitata, ottenendo cosí una memoria RAM da 4 Kbyte.

In tal caso, però, non si potranno piú immettere dati attraverso la tastiera, né visualizzarli sullo schermo.

Il blocco di memoria RAM da 4 Kbyte, locazioni da CØØØ (49152) a CFFF (53247), viene utilizzato per la programmazione del computer in codice macchina.

L'interprete BASIC è invece contenuto nelle locazioni di memoria che vanno da AØØØ (4Ø96Ø) a BFFF (49151); la sua funzione è di tradurre le istruzioni BASIC in codice macchina.

Pure quest'area può essere disattivata per modificare il linguaggio BASIC residente oppure per utilizzare un altro linguaggio di programmazione che viene memorizzato nella RAM da 8 Kbyte.

Infine, il blocco di memoria RAM posto dopo l'interprete BASIC costituisce l'area disponibile per la programmazione BASIC.

In questo blocco sono presenti particolari aree di memoria, che svolgono le seguenti funzioni.

— Uso delle cartucce: locazioni di memoria da 8ØØØ (32768) a 9FFF (4Ø959).

— Rappresentazione grafica ad alta risoluzione: indirizzi di memoria da 2ØØØ (8192) a 4ØØØ (16384).

— Posizionamento dei caratteri sullo schermo (area video): locazioni di memoria da 4ØØ (1Ø24) a Ø7E7 (2Ø23).

— Area di lavoro del sistema operativo: indirizzi di memoria da ØØØØ (Ø) a Ø3FF (1Ø23).

COMPUTER VIC-20

Nella *fig. 21* è riportata la mappa di memoria del computer VIC-20, nella quale le locazioni di memoria sono contraddistinte con numerazione esadecimale e decimale.

L'area di memoria le cui locazioni vanno da EØØØ a FFFF in esadecimale, ossia da 57344 a 65535 in decimale, contiene il sistema operativo che sovraintende al funzionamento del computer.

Il blocco di memoria successivo, compreso fra le locazioni CØØØ (49152) e DFFF (57343), contiene l'interprete BASIC, il cui compito, come è noto, è quello di tradurre le istruzioni BASIC in linguaggio macchina.

L'area di memoria da AØØØ (4Ø96Ø) a BFFF (49151) è costituita da un'espansione ROM, nella quale sono memorizzati i programmi in codice macchina.

La memoria colore del computer è contenuta nelle locazioni comprese fra 94ØØ (37888) e 95FF (38399).

Il blocco di memoria successivo, da 9ØØØ (36864) a 912F (37167), comprende le interfacce INPUT/OUTPUT e gli indirizzi del sistema.

L'area di memoria ROM da 4 Kbyte, le cui locazioni vanno da 8ØØØ (32768) a 8FFF (36863), viene utilizzata per la generazione dei vari caratteri visualizzabili sullo schermo.

A quest'area seguono tre blocchi di espansione di memoria RAM/ROM da 8 Kbyte a disposizione dell'utente.

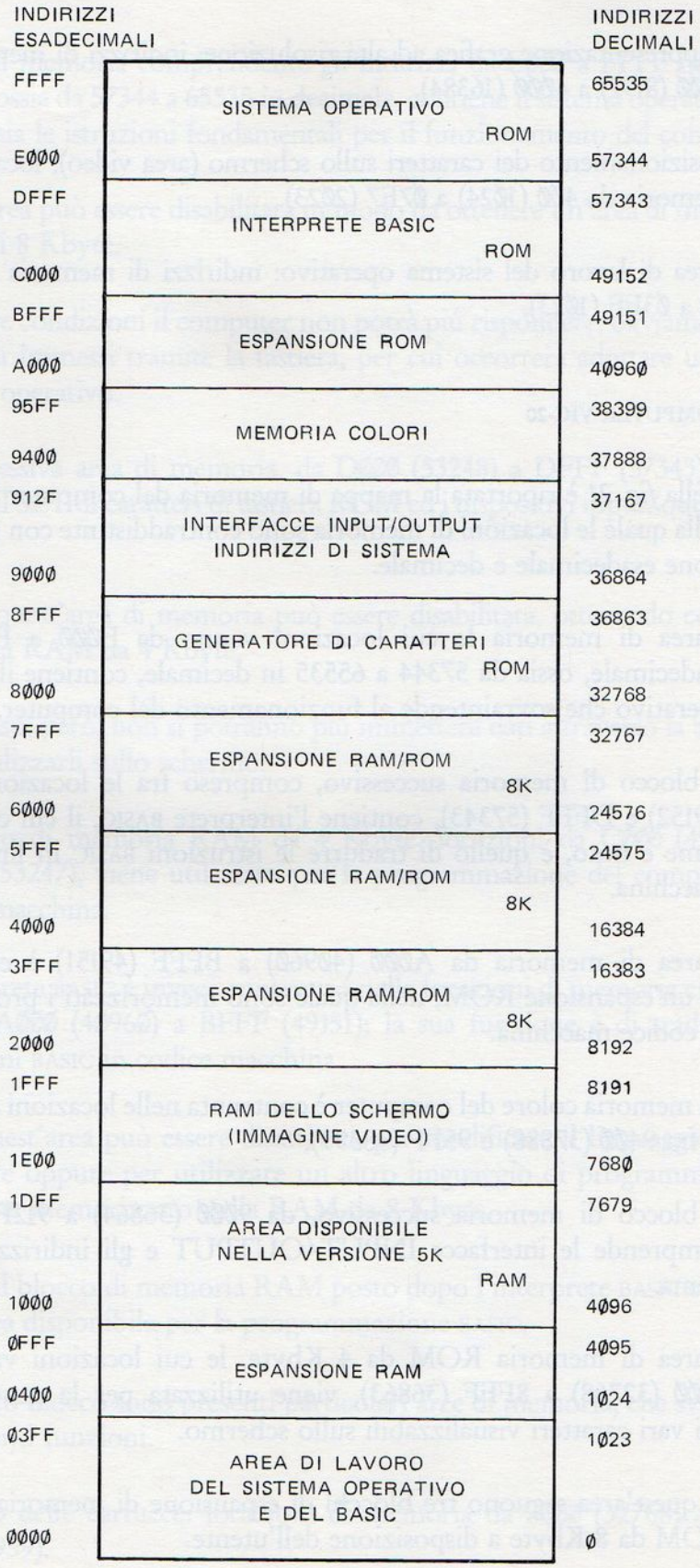


Fig. 21 – Mappa di memoria del VIC-20.

Le locazioni da 1EØØ (768Ø) a 1FFF (8191) costituiscono la memoria RAM dello schermo, nella quale viene memorizzata l'immagine da visualizzare.

L'area di memoria RAM successiva, da 1ØØØ (4Ø96) a 1DFF (7679), è quella a disposizione dell'utente per la programmazione quando viene usato il computer nella versione standard, ossia con 5 Kbyte di memoria.

Le locazioni di memoria da Ø4ØØ (1Ø24) a ØFFF (4Ø95) rappresentano un'espansione RAM a disposizione dell'utente per la programmazione BASIC.

Infine, nelle locazioni di memoria da ØØØØ(Ø) a Ø3FF (1Ø23) è prevista l'area di lavoro del sistema operativo e del BASIC.

* * * * * * *

CONTROLLO DIDATTICO 12

Domande

1) - Che cosa si intende per file d'informazioni?

.....

.....

.....

2) - Nei personal computer in quanti modi può essere organizzato un file d'informazioni?

.....

.....

.....

3) - Quale tipo di organizzazione dei file d'informazioni può essere utilizzato su un nastro magnetico e quale su un floppy-disk?

.....

.....

.....

.....

4) - Quali sono le operazioni preliminari per operare su un file-dati e quali sono quelle finali?

.....

.....

.....

.....

.....

.....

.....

.....

5) - Che cosa si intende per elenco "PISTA-SETTORE"?

.....

.....

.....

.....

.....

.....

6) - Come possono essere suddivisi un file sequenziale e un file ad accesso diretto?

.....

.....

.....

.....

.....

.....

.....

.....

7) - Illustrare il significato della seguente istruzione:

```
10 OPEN 2, 1, 1, "A"
```

.....

.....

.....

.....

8) - Quale significato hanno i parametri elencati dopo le istruzioni **PRINT #** e **INPUT #**?

.....

.....

.....

.....

9) - L'archivio dipendente esemplificato nel programma NOTA SPESE eseguito con la lezione *Pratica del BASIC 12* può essere generalizzato per altre aziende. Che cosa occorrebbe fare per raggiungere tale scopo?

.....

.....

.....

.....

.....

.....

.....

.....

10) - Quale errore è contenuto nel segmento di programma che segue?

```
10 OPEN 2,1,1,"DATI"
20 FOR X=1 TO 10
30 PRINT#2,X
40 NEXT X
50 CLOSE 3
```

Risposte

1) - Un file d'informazioni è un flusso organizzato di dati (generalmente non omogeneo, cioè costituito da dati numerici e alfanumerici).

2) - Nei sistemi personal computer i file d'informazioni possono essere organizzati in modo sequenziale ed in modo casuale o ad accesso diretto.

3) - Su un nastro magnetico può essere utilizzata soltanto l'organizzazione sequenziale; sul disco, oltre all'organizzazione sequenziale, può essere usata anche quella casuale.

4) - Le operazioni iniziali per accedere ad un file-dati si condensano nell'istruzione di apertura **OPEN**, seguita da un elenco di parametri ed opzioni (tra cui il nome del file-dati). Le operazioni finali consistono nell'istruzione **CLOSE**, seguita generalmente dal numero di buffer associato al file precedentemente aperto. Al posto di questo numero può esservi soltanto il nome del file.

5) - Un elenco "PISTA-SETTORE" è un particolare file, registrato su disco, che serve come guida per la ricerca veloce delle informazioni di un file-dati sul disco stesso. L'elenco pista settore può essere considerato come il sistema di coordinate per l'individuazione immediata dei dati sul disco.

6) - Un file sequenziale si può soltanto suddividere in campi. Un file ad accesso diretto si può suddividere in record, ognuno dei quali a sua volta si suddivide in uno o piú campi. La lunghezza totale di ogni record è fissa e viene specificata nei parametri dell'istruzione **OPEN**.

7) - L'istruzione provvede ad aprire il file numero 2 sull'unità a nastro per la registrazione del file di nome A.

8) - Il numero riportato dopo le istruzioni **PRINT #** (registrazione) e **INPUT #** (lettura) fa riferimento al numero di buffer associato al file-dati. Questa associazione viene stabilita con l'istruzione **OPN**.

9) - Innanzitutto l'archivio riportato tra i numeri di riga 21Ø e 42Ø del programma "CREAZIONE DATA-DIP" rappresentato nella lezione *Pratica del* BASIC *12* può essere esteso aggiungendo altre istruzioni DATA al termine del listato.
L'archivio cosí ampliato deve riportare al termine i codici dipendenti (artificiali) 999 e 1ØØØ seguiti da due *, uno Ø ed AØ, cosí come indicato tra i numeri di programma 38Ø e 42Ø.
Il successivo programma NOTA SPESE comprenderà istruzioni **DATA** contenenti i codici relativi al nuovo archivio ampliato.
In ogni caso si possono modificare numero di dipendenti, nominativi, indirizzi, codici, ecc.

10) - Nel segmento di programma indicato risulta errata l'istruzione 5Ø, in quanto il numero del file che si desidera chiudere (3) non corrisponde a quello di apertura (2).

* * * * * * *

PARLA BASIC

Corso teorico-pratico per corrispondenza di

ANALISI E PROGRAMMAZIONE BASIC

Scuola Radio Elettra

10126 Torino - Via Stellone 5

Tel. (011) 674432